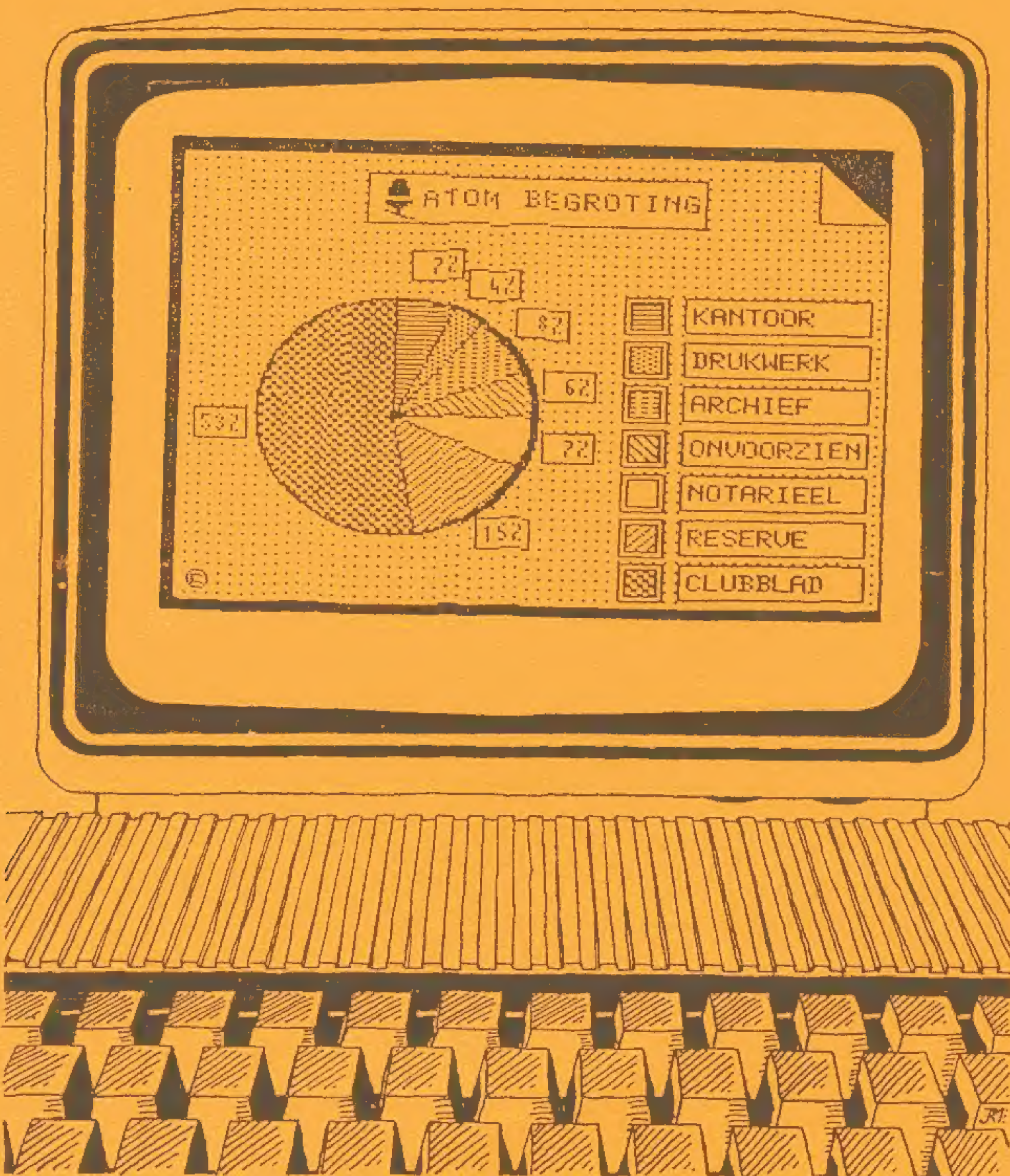
ACORN NIEUWS

JAARGANG

NUMMER



Bestuur:

Voorzitter:
N. Stad
Plataanweg 47.
1544 PB Zaandijk.
Telf. 075-280808

Secretaris:
N. Schoone
Sluispad 55-B
1521 EZ Wormerveer
Telf. 075-286624

Penningmeester:
Th. van Kempen.
Het Puyven 71.
5672 RB Nuenen.
Telf. 040 - 036210

Clubwinkel:
N. Stad
Plataanweg 47
1544 PB Zaandijk
Telf. 075 - 280808

Hard-ware cie.:
P. Ehrlich
Roostenlaan 266
5644 BG Eindhoven.
Telf. 040 - 114183

Redaktie Acorn Nieuws.:
H.Reinders.
Leeuwarderstraat 8.
9718 HX Groningen.
Telf. 050 - 125450

Contributie 1985: fl. 60.00
Leden buitenland fl. 75.00

Giro 5244293 BANK 52.84.69.010
Beide t.n.v. Penningmeester Acorn computerclub.
Nuenen.

Redactie ACORN NIEUWS:
Rudi van Drunen - hardware
Hans Marks - software
Henk Reinders - layout

Redactie adres:
Postbus 1050
9700 BA Groningen
Telf. 050-125450

Ledenadministratie:
S. van Leeuwen
Kompasstraat 32.
1973 PX IJmuiden
Telf 02550-22435

Datasheets:
G. Akkermans.
Wikke 1
1273 DR Huizen.
Telf 02152 - 60294.

Drukwerkarchief:
F. Monsato
Biesbosch 153
8032 VD Zwolle.

Uiterste datum inlevering copy:
NR 4. 19 - 4 - 1985.
NR 5. 5 - 7 - 1985.
NR 6. 16 - 8 - 1985.
NR 7. 27 - 9 - 1985.
NR 8. 8 - 11 - 1985.

DE CLUB-WINKEL:

Geheugenkaart; 16 kByte extra in de ATOM	fl. 40,00
Schakelkaart; meerdere EPROM's op Axxx (incl.74LS133)	fl. 50,00
Programmerkaart; zelf programmeren van EPROMs	fl. 21,00
Big Benny print;incl. Clock IC MSM 5832RS	fl. 42.50
Los IC MSM 5832RS	fl. 32.50
Minischakelkaart	fl. 16.00
Herdruk ACORN NIEUWS 1982; 97 pag. wetenswaardigheden	fl. 6.00
Jaargang 1983; totaal ruim 450 pag.	fl. 30.00
Jaargang 1984;	fl. 35.00
ATOM-WARE deel 1; machinetaal op de ATOM 98 pag.	fl. 6.00
ATOM-WARE deel 2; het diskdrive operatingsystem 68 pag.	fl. 5.00

Bestellen:
Bij Uw regionale penning-meester.
Rechtstreeks bij de federatieve penningmeester, dan fl. 4.00
extra voor porto-kosten.

ACORN NIEUWS is een uigave van de Fed. Acorn computerclubs n/b
verschijnt 6 - 8 keer per jaar.

De redaktie gaat er vanuit, dat ingezonden copij afkomstig is van de inzender, tenzij uitdrukkelijk is vermeld.
Het is ondoenlijk voor elke inzending nate gaan of het ontwerp uit een andere publicatie afkomstig is.
De aansprakelijkheid voor auteursrechten voor ingezonden copy ligt dus bij de inzender.

DE PRINTEN.

De schakelkaart is vanaf eind april wederom leverbaar. Voor de minischakelkaart en de epromprogrammerkaart geldt leverbaar zolang de voorraad strekt.

ADRESWIJZIGINGEN:

Adreswijzigingen dienen rechtstreeks schriftelijk gestuurd te worden aan SIMON VAN LEEUWEN de ledenadministrateur.

VOORPLAATJE

Het plaatje op de voorkant is deze keer afkomstig uit het programma Pie-chart voor gags.

INHOUD REGIO-SCHIJF NR. 3

SHELDBS	2900	AFAF	016A3	002	A.N. 1985 nr. 3 blz. 9
BOX	2900	AFAF	002C9	019	A.N. 1985 nr. 3 blz. 93
SCRBACK	8200	C2B2	00693	01C	A.N. 1985 nr. 3 blz. 16
TELETAB	8200	C2B2	003E0	023	A.N. 1985 nr. 3 blz. 21
PIE-CHA	2900	AFAF	0219B	027	A.N. 1985 nr. 3 blz. 22
MENU-1	2900	AFAF	0219B	049	A.N. 1985 nr. 3 blz. 30
MENU-2	2900	AFAF	003B2	06B	A.N. 1985 nr. 3 blz. 30
DRUM-S1	2900	AFAF	0168F	06F	A.N. 1985 nr. 3 blz. 30
DRUM-VI	A000	A000	01000	086	A.N. 1985 nr. 3 blz. 30
ANALYSE	2900	AFAF	00B35	096	A.N. 1985 nr. 3 blz. 38
TORTRAC	2900	AFAF	00306	0A2	A.N. 1985 nr. 3 blz. 38
SCHSOFT	2900	AFAF	0094A	0A6	A.N. 1985 nr. 3 blz. 46
SCREENR	A000	A000	01000	0B0	A.N. 1985 nr. 3 blz. 50
INMAKER	2900	AFAF	007BA	0C0	A.N. 1985 nr. 3 blz. 54
PLANTAR	2900	CE86	0534F	0C8	A.N. 1985 nr. 3 blz. 66
DISKCAT	2900	AFAF	006D9	11C	A.N. 1985 nr. 3 blz. 70
PCAT2	2900	AFAF	00310	123	A.N. 1985 nr. 3 blz. 70
PCAT1	2900	AFAF	001B8	127	A.N. 1985 nr. 3 blz. 70
DFCAT	8200	AFAF	007EA	129	A.N. 1985 nr. 3 blz. 70
PC-SORT	8200	AFAF	002FC	131	A.N. 1985 nr. 3 blz. 70
NECDUMP	2900	AFAF	006F9	134	A.N. 1985 nr. 3 blz. 82
DIKTEE	2900	AFAF	006ED	13B	A.N. 1985 nr. 3 blz. 91
VERTAAL	2900	AFAF	0052D	142	A.N. 1985 nr. 3 blz. 92
ELISE	2900	AFAF	005DA	148	A.N. 1985 nr. 3 blz. 95
300	65C0	36E5	00200	14E	A.N. 1985 nr. 3 blz. 62
301	65C0	36E5	00200	150	A.N. 1985 nr. 3 blz. 62
DATOMIX	2900	AFAF	00890	152	A.N. 1985 nr. 3 blz. 96
BASICOD	2900	AFAF	010F9	15B	is nog geen beschrijving van.

HOERA HOERA ! ! !

In tegenstelling met eerdere berichten, het IC 9345, is in een vernieuwde versie reeds in ons bezit. De ontwikkeling van de 80-koloms-kaart gaat nu in volle hevigheid verder.

10 APR 1985 TWENTE MEPA-GEBOUW.		
20.00 UUR	WIERDENSESTRAAT 40	ALMELO.
11 APR 1985 NOORD MUNSTERHOES		
19.30 UUR	MUSTERHEERD.	GRONINGEN.
11 APR 1985 CENTRUM DE LANTAERN		
20.00 UUR	UTRECHTSESTRAATWEG 4	NIEUWEGEIN.
13 APR 1985 NOORD-H DE KOPEREN KNOOP		
12.00 UUR	VAN LIMBURG STIRUM STRAAT	AMSTERDAM.
16 APR 1985 BRABANT DE WERFF		
20.30 UUR	V/D WERFFSTRAAT	EINDHOVEN.
17 APR 1985 NOORD M van VLISSINGEN		
	ACHTER DE GROTE KERK 25	LEEUWARDEN.
20 APR 1985 BELGIE OXACO-CAPENBERGCENTER		
10.30 UUR	BORSBEEKSE STEENWEG 45.	BOECHOUT (BIJ HOVE)
26 APR 1985 DEN HAAG EXODUSKERK		
20.00 UUR	BERESTEINLAAN 263	DEN HAAG.
2 MEI 1985 BRABANT GEBOUW B.R.A.N.		
20.30 UUR	HEUVEL 7	VORSTENBOSCH.
6 MEI 1985 ARNHEM T.O. POSTGIRO		
20.00 UUR	VELPERWEG 56-A (kelder)	ARNHEM
8 MEI 1985 TWENTE MEPA-GEBOUW.		
20.00 UUR	WIERDENSESTRAAT 40	ALMELO.
8 MEI 1985 CENTRUM DE LANTAERN		
20.00 UUR	UTRECHTSESTRAATWEG 4	NIEUWEGEIN.
9 MEI 1985 NOORD DE TREF-KOEL		
19.30 UUR	ZONNELAAN.	GRONINGEN.
11 MEI 1985 NOORD-H DE KOPEREN KNOOP		
12.00 UUR	VAN LIMBURG STIRUM STRAAT	AMSTERDAM.
15 MEI 1985 NOORD M van VLISSINGEN		
	ACHTER DE GROTE KERK 25	LEEUWARDEN.
21 MEI 1985 BRABANT DE WERFF		
20.30 UUR	V/D WERFFSTRAAT	EINDHOVEN.
31 MEI 1985 DEN HAAG EXODUSKERK		
20.00 UUR	BERESTEINLAAN 263	DEN HAAG.

Regio's, die hier niet vermeld staan, hebben geen opgaven naar de redactie gestuurd.

DOCUMENTATIE ATOM-MERGE

1. Algemeen
   - Met dit programma is het mogelijk om 2 programma's samen te voegen (mergen).
   - De mogelijkheid bestaat ook voor 2 bestanden mits het regelnummer als recordkey gebruikt wordt.
   - Als enigste toolbox wordt de 'P-CHARME' gebruikt.

2. Programmawerking
   - Het programma vraagt om de adressen van de samen te voegen programma's of bestanden. Optioneel is het ingeven van een bepaalde serie regelnummers die met het tweede programma (of een serie regelnummers daarvan) samengevoegd dienen te worden. Men kan ook volstaan met het ingeven van 'CR'.
   - Als in de samen te voegen programma's dezelfde regelnummers voorkomen wordt een error 400 gegenereerd en is het programma alleen nog met 'ESC' af te breken.
   - Het statement 'END', al of niet met spaties en/of afgekort, moet zich als enigste statement op de laatste regel van het programma bevinden.
   - Tevens dient men er zelf op te letten dat voor alle programma's (atom-merge, A en B) of programma en bestanden voldoende geheugenruimte beschikbaar is.

3. Programmavariabelen

```
- Voor 'programma A'      :  A = programma adres
                             B = adrespointer regelnummer
                             C = regelnummer
                             I = einde programma indikator
                            $D = statement
                            $F = regelnummer va
                            $G = regelnummer tot en met
- Voor 'programma B'      :  V = programma adres
                             W = adrespointer regelnummer
                             X = regelnummer
                             J = einde programma indikator
                            $Y = statement
                            $S = regelnummer van
                            $T = regelnummer tot en met
- Voor 'nieuwe programma' :  K = programma adres
                             L = adrespointer regelnummer
                            $N = statement
- Algemeen                 :$Z = inputvariabele
```

4. Listing

```
 10 PROGRAM ATOM-MERGE
 20 REM AUTEUR
 30 REM H.LINSSEN
 40 REM ROERMOND
 50 REM 04750-27579
 60 REM VERSIE 1.0 010984
 70 REM TOOLBOX:P-CHARME
 80
 90 PROC FOOTNOTE
100  IF I<1 C=?B*256+B?1
```

```
110  IF J<1 X=?W*256+W?1
120  VTAB15
130  P.C"    "X
140 PEND
150
160 PROC COPY-D
170  FOOTNOTE
180  D=B+2
190  IF C=VALG I=1
200  IF $D=" E." OR $D=" END" OR $D="E." OR $D="END" I=1
210  IF I=1 IF J=0 GOTO 280
220   ?L=?B
230   L?1=B?1
240   N=L+2
250   $N=$D
260   L=L+LEN(N)+3
270   B=B+LEN(D)+3
280 PEND
290
300 PROC COPY-Y
310  FOOTNOTE
320  Y=W+2
330  IF X=VALT J=1
340  IF $Y=" E." OR $Y=" END" OR $Y="E." OR $Y="END" J=1
350  IF J=1 IF I=0 GOTO 420
360   ?L=?W
370   L?1=W?1
380   N=L+2
390   $N=$Y
400   L=L+LEN(N)+3
410   W=W+LEN(Y)+3
420 PEND
430
440 PROC ERROR
450  VTAB 14
460  P.$7
470  P."ERROR 400  "'
480  P.C"    "X" EQUAL LINE NUMBERS"
490 PEND
500
510 REM init
520 DIM Z1,F4,G4,S4,T4
530 P.$12;P."       ATOM-MERGE"'
540 P."================================"
560 I=0
570 @=0
580 J=0
590 IN.'"ADDRESS PROGRAM A: "A
600 IN."FROM LINE-NUMBER : "$F;IF $F="" $F="0"
610 IN."  TO LINE-NUMBER : "$G;IF $G="" $G="99999"
620 IN.'"ADDRESS PROGRAM B: "V
630 IN."FROM LINE-NUMBER : "$S;IF $S="" $S="0"
640 IN."  TO LINE-NUMBER : "$T;IF $T="" $T="99999"
650 IN.'"DESTINATION ADDRESS: "K
660 IN.''"          OKAY N/'CR'"$Z
670 IF $Z="N" GOTO 530
680 P.$12;?#E1=0
690 P."       ATOM-MERGE"'
700 P."================================"
```

```
710 P.''''"MERGE OF PROGRAM AT #"&A'
720 P."       AND PROGRAM AT #"&V'
730 P."  TO NEW PROGRAM AT #"&K'
740
750 REM MERGE
760 B=A+1
770 W=V+1
780 ?K=#0D
790 L=K+1
800 VTAB 14;P."BEGIN MERGE"
810aC=?B*256+B?1
820 IF C<VALF D=B+2;B=B+LEND+3;G.a
830 X=?W*256+W?1
840 IF X<VALS Y=W+2;W=W+LENY+3;G.830
850 IF C=X ERROR
860 IF C<X;COPY-D
870 IF X<C;COPY-Y
880 IF I=1;DO COPY-Y;UNTIL J=1;G.910
890 IF J=1;DO COPY-D;UNTIL I=1;G.910
900 GOTO a
910 ?L=#FF
920 ?#E1=128
930 P.$7"   READY"
999 END
```


DE PRINTER.

Daar ik nog niet zo lang geleden een printer op m'n "micro" heb aangesloten,wil ik je daar-over eenz wat meer vertellen.

Een prachtig inztrument.Men kan het eigenlijk in de moderne maatschappij niet meer missen.Onlangs heb ik me duz zo'n apparaat aangezschaft.Ik kon het voor een prikje op de kop tikken op het Waterlooplein,voor een belachelijk lage prijz.

Tien gulden iz werkelijk niet veel voor zo'n prachtig mechaniek.Welizwaar doet een lettertje het niet,namelijk de Z.Dat maakt niet zoveel uit,het is nog best te lezen,nietwaar? Alz er dan maar niet meer gaat haperen dan iz het niet zo elg.......

Veldlaaid daal heb je het al,de L van lotzooi iz uitgevallen.Nu iz dat niet zo elg,want daal gebluik ik gewoon de L maal vool.Je kunt toch ook wel een beetje chinees lezen? Ja,het iz welkelijk een genoegen zo'n plachtig inztlument te bezitten.Het gaat veel vluggel dan zghlijven.

He,dat iz niet zo leuk,daal doet de G van Golumbuz het ook niet meel.Ik moet het toestel togh maal eenz nakijken.Want zo gaat tet.....wel veldulleme,daal gaat de T van Tendlik. Tij zit muulvast.Ik begim (alweel een) togt te gelovem dat deze magtime miet zo bezt iz.

Il moezt tet de tamdelaal maal eenz vlagen.Il begim nu togt te gelovem dat il bedlogem bem,wamt tet megimt mu togt te elg te woldem.Tet iz velzghtlillllelijk......op deze mamiel zgu je zelf miet meel lummen lezem wat je zgtlijft.

Em dbbl tgef je mbtuullijk geem plimtel vggl te lgpem. Ubm lum je met zg gted mat de tmbd zgtlijvem,dbt ggl iz miet te lezem.

SHELOBS ARMY

Onder het motto: "You've read the book, seen the motion picture....now play the game!", volgt hier de lancering van een spel, gebaseerd op een hoofdstuk uit het fantastische boek 'In de ban van de ring' door J.J.R. Tolkien.
Dit boek beschrijft de reis van Frodo en zijn reisgenoten door het imaginaire land Midden-Aarde naar het donkere land Mordor om daar de ring, met bijzondere krachten, in het vuur te gooien waarin het gesmeed is, om zodoende de zwarte kracht van een onguur persoon genaamd Sauron, te vernietigen.
Shelobs Army is een spel wat zeer luchtig gebaseerd is op het verhaal wat beschreven is in hoofdstuk 9, vierde boek, deel 2.
Dit hoofdstuk beschrijft de moeizame tocht door de lange gang die bewaakt wordt door de grote spin Shelob. In dit spel wordt Shelob voorgesteld door een aantal spinnen die de verschillende gangen bewaken.
Noodzakelijk voor dit spel zijn:

- GAGS V2.0E
- een joystick

Wanneer men vanuit de startpositie Frodo naar boven wil geleiden moet eerst de ring gebruikt worden. Dit kan d.m.v. het indrukken van de fire-button en een opwaartse beweging van de joystick.
D.m.v. de ring is het nu ook mogelijk om door de muren naar een andere gang te lopen. Elke stap naar boven levert 5 punten op doch elke stap naar beneden kost je een punt. Horizontale bewegingen van Frodo (fire-button indrukken niet noodzakelijk) kost- of levert geen punten.
Eenmaal vanuit de startpositie vertrokken zal de energie van de ring afnemen. Als deze 0 is, ben je ten dode opgeschreven.
Let op: eenmaal in een gang aanwezig, zal de spin ruiken dat je er bent en op je af komen, dus maak dat je wegkomt.
Genoeg gepraat...spelen!

```
 10 REM ////////// SHELOB'S ARMY \\\\\\\\\\\\
 20 REM / GEBASEERD OP 'LORD OF THE RINGS'  \
 30 REM /  EEN TRILOGIE DOOR J.J.R. TOLKIEN \
 40 REM NODIG: GAGS-V2.0E
 50 REM         JOYSTICK
 60 REM AUTEUR: GERRIT HILLEBRAND
 70
 80 BASE((TOP/256)+2);GOS.i;REM initiate sprites
 90 DIM T(15),RR(5)
100
110 GOS.o;REM information text
120 W=0
130 E=50;U=8;S=0;C=0
140 GOS.d;REM draw screen
150 SET:FRODO,125,38
160
170 REM MAIN LOOP
180 I=-1
190 DO
200  I=I+1;IFI=6;I=0;V=0
210  POS(10+I),P,Z;IF P+RRI>229;RRI=-RRI;SHOVE(10+I),RRI,0;G.c
220  IF P+RR(I)<10;RRI=-RRI;SHOVE(10+I),RRI,0
230c SHOVE(10+I),RRI,0
240  POS:FRODO,X,Y
250  IF Y>=180;GOS.w
```

```
 260  ATHIT:FRODO,(I+10);GOS.h
 270  IF ABS(Y-Z)<8;U=16;RRI=U*((P<X)-(P>X))
 280  IF V;G.f
 290  IF Y>40;GOS.e
 300  G=0;H=0;JOYSTK G,H,Z
 310  IF Z=0;G.g
 320  V=1
 330  IF Y=38;IF H<>0;IF H<>1;G.g
 340  IF H=1;IFZ<>0;SHOVE:FRODO,0,6;S=S+5;GOS.s;G.g
 350  IF H<>0;IFZ<>0;SHOVE:FRODO,0,-6;S=S-1;GOS.s
 360g IF G=0;G.f
 370  V=1
 380  IF G=1;IF X+U>229;G.f
 390  IF X-U<10;G.f
 400  SHOVE:FRODO,((G=1)*6-(G<>1)*6),0
 410fU.N=0 OR E=0
 420 FOR I=1 TO 8
 430  SOUND (I*10),200
 440  MODE I
 450  PAUSE 10
 460 NEXT
 470 CLS
 480 @=4
 490 P.''''"YOUR SCORE WAS "S'
 500 IF S>W;W=S
 510 P."HIGH SCORE "W'
 520 P.'"PRESS KEY TO PLAY "
 530 LINK#FFE3;G.130
 540 END
 550
 560hIMAGE:FRODO,(190+20*(N-1)),18
 570 N=N-1
 580 CARRY:FRODO,120,38
 590 INV
 600 F.J=40TO80
 610  SOUND J,30
 620 NEXT
 630 INV
 640 GOS.l
 650 RETURN
 660
 670eE=E-1
 680 BLOCK2,(11+50-E),12,1,10
 690 RETURN
 700
 710tREM DRAW MESSAGES
 720 J=0;B=8*LEN(T)-1
 730  FOR I=X TO X+B STEP 8
 740    IMAGE (T?J),I,(Y+6)
 750    J=J+1
 760  NEXT I
 770 RETURN
 780
```

```
 790sREM UPDATE SCORE
 800 GOS.r;C=S
 810rJ=(C%10000)/1000+48
 820 K=(C%1000)/100+48
 830 L=(C%100)/10+48
 840 M=C%10+48
 850 IMAGE J,138,18
 860 IMAGE K,148,18
 870 IMAGE L,158,18
 880 IMAGE M,168,18
 890 RETURN
 900
 910iDEF FRODO,0000000111111000
 920 DEF:      0000011111111100
 930 DEF:      0000111111100110
 940 DEF:      0000110011110011
 950 DEF:      0001111111110011
 960 DEF:      1111111011110001
 970 DEF:      0001100111000001
 980 DEF:      0000011110000000
 990 ASSIGN FRODO,16
1000 FORI=10 TO 15
1010  CREATE:SPIDER,#98,#85,#7F,7,15,#15,#24,#CC,#19,#A1,#FE,#E0,#F0,#A8,#24,#33/A:
1020 NEXT I
1030  CREATE SPACE,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0/A:CH" "
1040  CREATE ZERO,0,#70,#88,#C8,#A8,#98,#88,#70,0,0,0,0,0,0,0,0/A:48
1050  CREATE ONE,0,#70,#20,#20,#20,#20,#60,#20,0,0,0,0,0,0,0,0/A:CH"1"
1060  CREATE TWO,0,#F8,#80,#80,#70,#08,#88,#70,0,0,0,0,0,0,0,0/A:CH"2"
1070  CREATE THREE,0,#70,#88,#08,#30,#08,#88,#70,0,0,0,0,0,0,0,0/A:CH"3"
1080  CREATE FOUR,0,#10,#10,#F8,#90,#50,#30,#10,0,0,0,0,0,0,0,0/A:CH"4"
1090  CREATE FIVE,0,#70,#88,#08,#08,#F0,#80,#F8,0,0,0,0,0,0,0,0/A:CH"5"
1100  CREATE SIX,0,#70,#88,#88,#F0,#80,#40,#30,0,0,0,0,0,0,0,0/A:CH"6"
1110  CREATE SEVEN,0,#80,#80,#40,#20,#10,#08,#F8,0,0,0,0,0,0,0,0/A:CH"7"
1120  CREATE EIGHT,0,#70,#88,#88,#70,#88,#88,#70,0,0,0,0,0,0,0,0/A:CH"8"
1130  CREATE NINE,0,#60,#10,#08,#78,#88,#88,#70,0,0,0,0,0,0,0,0/A:CH"9"
1140  CREATE COLON,0,0,#60,#60,0,#60,#60,0,0,0,0,0,0,0,0,0/A:CH":"
1150  CREATE A,0,#88,#88,#88,#F8,#88,#88,#70,0,0,0,0,0,0,0,0/A:CH"A"
1160  CREATE B,0,#F0,#48,#48,#70,#48,#48,#F0,0,0,0,0,0,0,0,0/A:CH"B"
1170  CREATE C,0,#70,#88,#80,#80,#80,#88,#70,0,0,0,0,0,0,0,0/A:CH"C"
1180  CREATE E,0,#F8,#80,#80,#F0,#80,#80,#F8,0,0,0,0,0,0,0,0/A:CH"E"
1190  CREATE H,0,#88,#88,#88,#F8,#88,#88,#88,0,0,0,0,0,0,0,0/A:CH"H"
1200  CREATE L,0,#F8,#80,#80,#80,#80,#80,#80,0,0,0,0,0,0,0,0/A:CH"L"
1210  CREATE M,0,#88,#88,#88,#A8,#A8,#D8,#88,0,0,0,0,0,0,0,0/A:CH"M"
1220  CREATE O,0,#70,#88,#88,#88,#88,#88,#70,0,0,0,0,0,0,0,0/A:CH"O"
1230  CREATE R,0,#88,#90,#A0,#F0,#88,#88,#F0,0,0,0,0,0,0,0,0/A:CH"R"
1240  CREATE S,0,#70,#88,#08,#70,#80,#88,#70,0,0,0,0,0,0,0,0/A:CH"S"
1250  CREATE Y,0,#20,#20,#20,#50,#88,#88,#88,0,0,0,0,0,0,0,0/A:CH"Y"
1260  CREATE QUOTE,0,0,0,0,0,0,#40,#40,0,0,0,0,0,0,0,0/A:CH"'"
1270  CREATE/P:1 #44,#11,#44,#11,#44,#11,#44,#11
1280 RETURN
1290
1300dCLEAR4
1310 BORDER0,0
1320 FOR I=1 TO 6;CUBE1,0,9,(44+(I-1)*22),235,18;NEXT
1330 CUBE1,0,4,4,247,24;CUBE1,0,4,40,247,134
1340 $T="SHELOB'S ARMY";X=80;Y=178;GOS.t
1350 $T="SCORE:";X=80;Y=12;GOS.t
```

```
1360 GOS.l
1370 FOR I=44TO164 S.20;PAINT 8,I,1;PAINT248,I,1;N.
1380 INV
1390 GOS.r
1400 BLOCK 2,6,6,243,20
1410 FOR I=0 TO 2;IMAGE:FRODO,(190+20*I),18;NEXT I
1420 N=3
1430 FOR I=0 TO 5;X=ABSRND%100
1440 SET(10+I),(120+X),(22*I+55);IF X<0;RR(I)=-U;G.n
1450 RR(I)=U
1460nNEXT
1470 RETURN
1480
1490wUNSET:FRODO;S=S+50;GOS.s
1500 SET:FRODO,125,38;U=U+2;IFU>16;U=16
1510 BLOCK0,10,8,54,16;GOS.l
1520 RETURN
1530
1540lE=50
1550 CUBE1,0,10,8,54,16;BLOCK1,12,12,50,10
1560 RETURN
1570
1580oREM UIT: TOLKIEN LEXICON - J.E.A. TYLER
1590 REM: BESCHRIJVING SHELOB'S LEGER
1600 CLS
1610 P."shelobs leger"'
1620 P."HET HOL VAN DE GROTE SPIN SHELOB"'"HET WAS EEN GROT VAN ENORME "
1630 P."AFMETINGEN, DOOR MIDDEL VAN GANGEN  EN TUNNELS MET ANDERE GROTTEN"'
1640 P."VERBONDEN EN HIJ LAG VLAK ONDER DE TOP VAN DE PAS VAN CIRITH" "UNGOL "
1650 P."AAN DE WESTZIJDE. DE WEG  OVER DE PAS WAS OP DIT PUNT DOORHARDE "
1660 P."ROTS GEGRAVEN EN REIZIGERSWAREN DIENTENGEVOLGE GEDWONGEN"'
1670 P."DE BUITENSTE GANGEN VAN HET"'"LEGER VAN DE SPIN (TORECH UNGOL)TE"
1680 P."GEBRUIKEN OP HUN WEG."'
1690 LINK#FFE3
1700 CLS
1710 P."IN DIT SPEL BENT U DE HOOFD PER-SOON UIT HET EPOS -IN DE BAN VANDE RING-"
1720 P." GENAAMD FRODO. OM DOOR DE GANGEN VAN SHELOB TE KOMEN"'"MOET U MET BEHULP"
1730 P." VAN EEN"'"JOYSTICK FRODO BESTUREN. NAAR   LINKS EN "
1740 P."RECHTS GEEFT GEEN ENER-GIE VERLIES"
1750 P." DOCH NAAR BOVEN EN  BENEDEN ZULT U OOK OP DE FIRE-  "
1760 P."BUTTON MOETEN DRUKKEN OM"
1770 P." DE GE- HEIMZINNIGE RING DIE U BIJ U    DRAAGT ENERGIE TE LATEN LEVEREN"
1780 P." OM DOOR DE GANGEN TE KUNNEN"'"LOPEN"'
1790 LINK#FFE3
1800 CLS
1810 P."INDIEN U FRODO HELEMAAL NAAR"'"BOVEN HEBT GELEIDT ZULLEN DE     "
1820 P."SPINNEN BIJ DE VOLGENDE RONDE"
1830 P."   SNELLER GAAN LOPEN. HET SPEL IS TEN EINDE INDIEN "
1840 P."U ALLE FRODOS  OF ALLE ENERGIE"
1850 P." HEBT VER-BRUIKT."'"VEEL SUCCES....(OH ELBERETH)"'
1860 LINK#FFE3
1870 RETURN
1880
```

DISK-DUMP

Het Backup programma van Ronald Boers werkt bij mij niet goed, daarom heb ik maar zelf iets geschreven.
Het dumpt de hele schijf op band inclusief de directory zodat het zootje ook weer automatisch terug te dumpen is op schijf (het programma hiervoor komt nog).
Je hoeft niet met het eerste programma te starten: dit is voor het geval de band vol raakt en opnieuw gestart moet worden.
Het programma gebruikt de P-charme 1200 Baud routine maar kan ook voor de Josbox aangepast worden (r.70 en 220 COS1-FCOS). De programma's van de schijf worden geladen vanaf #3000; wie krap in geheugenruimte zit kan het dumpprogramma op pag.#22 zetten en de programma's laden vanaf #2500.
Hiervoor het adres op regel 160 en 200 aanpassen.

```
 10 LINK#E000
 20 P.$12$15"*** disk-tape dump ***"'''
 25 IN."vanaf prog.nr. (0-31)"C
 30 P.'"schijf plaatsen.recorder:opname"'';LINK#FFE3
 40 S=#21C;*DIR
 50 A=#2000;!S=!A;S!4=A!4
 55 S!8=A!256;S?12=A?260
 60 S?13=#D;P.$12$S'''
 65 IFC>0;G.100
 70 COS1;!#20C=#FAE5F96E
 80 !#208=#C278C278
 90 *SAVE"DIR"2000 2200
100 !#208=#FE94FE52
110 T=10;V=#100;@=2;P.C
115 B=A+?#2105-8*C
120 FOR P=B TO #2008 S.-8
125 IFT>32;T=10;C=C+3;P.'C
130 !S=!P;S!4=P!4
140 N=7;DO N=N-1;U.S?N>32
150 S?(N+1)=#D;P." "$S
152 DOP." ";U.C.>T;T=T+10
160 X=#80;!X=S;X!2=#3000;X?4=#FF
170 LINK#E000;LINK#E477
180 *DIR
190 X!2=P!V;X!4=P!(V+2)
200 X!6=#3000;X!8=X!6+P!(V+4)
210 !#208=#C278C278
220 COS1;LINK#FAE5
230 !#208=#FE94FE52
240 N.;P.$7$7$7''"KLAAR";END
```

BACKUP

Dit programma kopieert een hele schijf naar de cassette met 1200 baud en is bedoeld, zoals de naam al aangeeft, als backup voor de diskgebruiker. BACKUP is een verbeterde versie van het eerder gepubliceerde programma DISKCAS. De verbeteringen die in deze versie zijn aangebracht zijn:

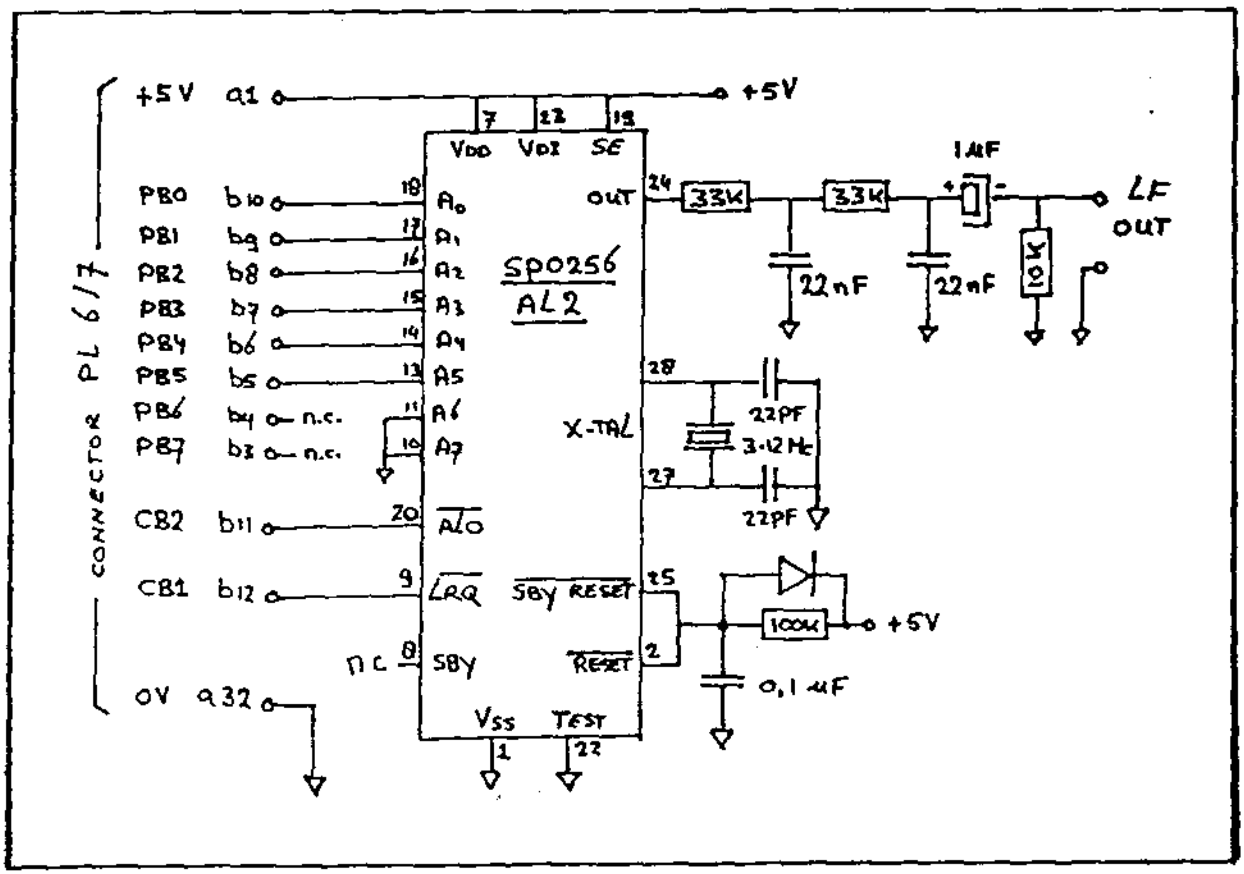
1. alle files worden geladen vanaf #2900; dus ook random-files en boxen kunnen nu gekopieerd worden.
2. ook files, die onder een andere dan de default-qualifier staan worden nu gekopieerd.
3. de files die het eerst op schijf gezet zijn, worden nu ook het eerst gekopieerd.

Aan het startadres kan men later, als men de file van tape af wil halen, zien of de file een BASIC-programma, een box, een random-access file of een tekstfile voor de editor is. Het programma moet gerund worden op #8200 om een optimaal gebruik te kunnen maken van het lage werkgeheugen.

```
10PROGRAM BACKUP
20REM VERSIE 1.2
30REM HELE DISK OP CASSETTE
40REM Frans le Blanc
50
60PROC TMENU
70M=0;Z=0;Y=TOP-36;S=TOP-57
80P.$12;CLEAR0;LINK#E000
90P."          BACKUP-PROGRAMMA"'
100P."        DISK NAAR CASSETTE"''
110P.$9"KOPIEERT HELE DISK  NAAR TAPE"'
120P.$9"ALLE FILES WORDEN VANAF #2900"'
130P.$9"GELADEN MET 1200 BAUD  (FCOS)"''
140P.$9"PLAATS DE DISKETTE IN DRIVE 0"
150MOVE0,40;DRAW62,40
160DRAW62,22;DRAW0,22
170DRAW0,40
180INKEYI;P.''
190*DIR
200*NOMON
210@=8;V=0;U=?#2105
220F=U+#2000;G=U+#2100
230F.N=0TO (U-8)S.8
240B=!(G+N)&#FFFF
250E=!(G+N+4)&#FFFF+B
260W=!(#210C+N)&#FFFF;V=V+W
270N.
280P."START RECORDER"'"DRUK OP EEN TOETS"'
290INKEYI
300P.'"PROGRAMMADUUR:        "
310P=V/1024*14/60+(U/8-Z)*10/60+1
320@=2;P.P" MINUTEN"'
330P."TOTALE GEHEUGEN:     "V/1024" KB"'
340P."AANTAL PROGRAMMA'S: "U/8-Z'
350PAU.200
360P.$12"              INDEX"';?#E1=0
370F.N=F TO #2008 S.-8
380P."    "
390F.I=0TO6;P.$(?(N+I));N.
400P."      "
410N.
420FCOS;REM uit diskbox
430*NOMON
440R=!#20A;?#20A=#93;?#20B=#FE
450*S."INDEX" 8000 8200
460!#20A=R
470PEND
480
490PROC DISKCOPY
500P.$12$15;?#E1=0
510DO;ADRES
520LINK#E000
530P."   ";F.N=0TO6;P.$(F?N);N.
540@=5;P." "&B,&E,&L'
550FLAAD;GOS.a
560PAU.200;PAU.200
570FSAVE
580P.$11"c"
590FCOS
600*NOMON
610R=!#20A;?#20A=#93;?#20B=#FE
620GOS.b
630!#20A=R;P.$11" "'
640F=F-8;G=G-8
650UNTIL F<#2008
660P.''"klaar"'
670P."STOP RECORDER"
680DO KEY I;BEEP 20,30;U.I<>0
690PEND
700
710PROC FLAAD
720X=Y;$X="*LOAD"""
730X=X+L.X;H=34
740F.N=0TO6
750XIF F?N=32;X?N=H;H=32
760ELSE X?N=F?N
```

```
770N.;X=X+7
780XIF H=34;?X=34
790ELSE ?X=32
800X=X+1
810$X=" 2900";X=X+L.X+1
820?X=?2;X?1=?1+1;X?2=CH"R";X?3=CH".";X?4=#D
830XIF F?7>#7F;?S=(F?7)-#80
840ELSE ?S=F?7
850PEND
860
870PROC FSAVE
880X=Y;$X="*SAVE";X?5=CH""""
890X=X+14
900Q=#2900;FADRES
910Q=E-B+#2900;FADRES
920Q=L;FADRES
930PEND
940
950PROC ADRES
960B=!G&#FFFF
970E=!(G+4)&#FFFF+B+1
980L=!(G+2)&#FFFF
990PEND
1000
1010PROC FADRES
1020I=1;X?0=#20
1030J=Q&#F000/#1000;ZADRES
1040J=Q&#0F00/#100;ZADRES
1050J=Q&#00F0/#10;ZADRES
1060J=Q&#000F;ZADRES
1070X=X+5
1080PEND
1090
1100PROC ZADRES
1110IF J>9;J=J+7
1120J=J+#30
1130X?I=J;I=I+1
1140PEND
1150
1160TMENU;DISKCOPY
1170BEEP 20,30;G.1160
1180
1190aREM SET QUALIFIER
1200*SETx
1210bREM LOAD/SAVE
1220*SAVE"xxxxxxx" 2900 EEEE LLLL
1230R.
```

SCHEMA VAN "STEM AAN DE ATOM" (zie pag. 87)



SCREEN BACKUP

Het onderstaande programma maakt het mogelijk om de inhoud van een scherm in textmode naar een bepaald geheugen te copieeren en terug. Basic programma's waarin men veel werkt met gegevens overzichten kunnen gebruik maken van deze routine. In de ASBK=I komt ongeveer een gelijksoortig statement voor, daar niet iedereen deze utility heeft en het basic programma toch universeel moet blijven is het aan te bevelen om zo'n routine vast in het programma op te nemen.

Het terughalen van eerder opgeslagen beeldschermen gebeurd nu met zeer hoge snelheid. Voor het gebruik van dit programma hoeft men geen kennis te hebben van de assembler omdat de routine helemaal in basic wordt geinitialiseerd en opgestart.

Heeft men een bestaand programma dat men wilt aanvullen met deze routine dan zal men eerst moeten bepalen waar de routine komt te staan en welk deel van het geheugen als backup moet worden gebruikt. De plaats voor de routine kan bijvoorbeeld achter de basic programma zijn. Men moet hierbij wel nog ruimte overlaten om het basicprogramma te kunnen uitbreiden. Indien men over laag-gestapeld geheugen beschikt dan kan de routine staan op bijvoorbeeld #3FA0 en zoniet dan op #3BA0.
Voor mensen met een disc-drive kan als backupgeheugen #2400 tot #2800 worden gebruikt (4 beeldschermen). Heeft men geen disc-drive of wilt men 8 beeldschermen opslaan dan kan het geheugen vnaf #8200 tot #9200 worden gebruikt.De selectie van de beeldschermen gebeurd door middel van een basic variabele, bijv. A=0;LINK#XXXX. De gebruikte variabele kan men zelf kiezen. Daarna wordt het startadres aangeroepen voor het "loaden" of "saven" naar het backup geheugen. Het aantal beeldschermen (afhankelijk van de behoefte en het geheugen) is 2, 4 of 8. Bij gebruik van 4 beeldschermen heeft het eerste scherm het nummer 0 en het vierde scherm nummer 3. De routine gebruikt alleen het van te voren bepaalde backup geheugen en zal dit niet overschrijden, ook als het beeldscherm nummer wordt overschreden.
Na het runnen van dit programma moet het startadres en het backup adres worden ingegeven. De routine heeft een lengte van #41 bytes. Voor elke 2 beeldschermen aan backup geheugen heeft men #400 bytes ofwel 1Kbyte aan vrij geheugen nodig. Vervolgens wordt er bepaald hoeveel beelschermen aanspreekbaar mogen worden. Hier worden alleen de getallen 2, 4 en 8 geaccepteerd. Vervolgens wordt er naar de variabele gevraagd die de beeldschermen moet selecteren. Bij voorkeur nergens anders in het basicprogramma gebruiken om problemen te voorkomen.
Men kan het assembleren via het beeldscherm en printer volgen door bij de volgende vragen "Y" of "N" in te typen.

Na het assembleren volgt de informatie omtrent de startadressen voor het "saven" en "loaden" van de beeldschermen. Hierbij is aan te raden om deze samen met topadres van de routine door middel van "REM" regels in het basic programma te vermelden, dit voorkomt veel zoekwerk bij eventuele latere wijzigingen.

```
 10 REM SCREEN BACKUP V3.1
 20 REM**********************
 30 REM*   CHARL DE MOOR    *
 40 REM*      JAN 1985      *
 50 REM*   045 - 2723 58    *
 60 REM**********************
 70 P.$6$12"SCREEN BACKUP V3.1"'"(C) CDM 1985"
 80 DIMLL30,A8,B8,V8;F.I=0TO10;LLI=#FFF;N.;@=0;?#E6=2
 90 IN.''"ASSEMBLING TO  :"Z
100 IN."BACKUP MEMORY  :"Y;LL9=Y;LL30=Y/256;IFZ=Y;G.70
110 IN."SCREENS (2,4,8):"$V;LL7=(CH$V)-49
120 IF LENV=1 A.CH$V=#32 ORCH$V=#34 ORCH$V=#38 G.140
130 P.$7$7$11;F.N=#80A0TO#80C0;?N=#20;N.;G.110
140 IN."VARIABLE (A-Z) :"$V;LL8=(CH$V)+737
150 IF LENV=1 A.CH$V>#40 A.CH$V<#5B G.170
160 P.$7$7$11;F.N=#80C0TO#80E0;?N=#20;N.;G.140
170 IN.'"LISTING (Y/N) :"$A
180 IN."PRINTER (Y/N) :"$B
190 P.$21;F.N=1TO2;P=Z;?#E8=0
200 IFN=2A.?A=89 P.$6''
210 IFN=2A.?B=89 P.$2'"SCREEN BACKUP V3.1"'"(C) CDM 1985"''
220[\START
230:LL0;\ SCREEN > BACKUP
240 JSRLL2;JMPLL4
250:LL1;\ BACKUP > SCREEN
260 JSRLL2;JMPLL3
270:LL2;LDY@#00;STY#8E;LDALL8;AND@LL7;CLC;ROLA
280 CLC;ADC@LL30;STA#8F;RTS
290:LL3;LDA(#8E),Y;STA#8000,Y;INC#8F
300 LDA(#8E),Y;STA#8100,Y;DEC#8F;INY;BNELL3;RTS
310:LL4;LDA#8000,Y;STA(#8E),Y;INC#8F
320LDA#8100,Y;STA(#8E),Y;DEC#8F;INY;BNELL4;RTS
330];N.;IF?#8E=#23G.400
340 P.$6"SCREEN BACKUP V3.1"'"(C) CDM 1985"'''
350 P." TOP : #"&Z'" END : #"&P'" SIZE:  ",(P-Z)" BYTES"''
360 P." SCREEN > BACKUP : #"&LL0'
370 P." BACKUP > SCREEN : #"&LL1''
380 P." BACKUP: #"&LL9''''
390 @=8;P.$3;END
400 P.$6$7$7$7''"OUT OF RANGE DETECTED !!"';G.390
410 ************************
415 * LABELS:              *
420 * LL7: NUMBER OF SCREEN *
430 * LL8: VAR. ZERO PAGE   *
440 * LL9: MEMORY BACKUP    *
450 ************************
460 * ZEROPAGE LOCATIONS:   *
470 * #8E,#8F: INDEX BACKUP *
480 ************************
```

HULPJES

Alvast veel plezier met het uitproberen van onderstaande "hulpjes".

- SNELLERE GOTO
- WHATLINE
- QUERY-QUERY
- BOOLE

SNELLERE GOTO

Het volgend truukje kan erg nuttig zijn, maar alleen wanneer:
- een programma groot genoeg is
- er veel GOTO's in zitten welke erg vaak gebruik moeten worden
- snelheid belangrijk is
- er rustig enkele extra bytes aan de programmatuur toegevoegd mogen of kunnen worden

De Atom kent twee sprong-opdrachten:
spring naar regelnummer
spring naar label
De eerste methode is relatief langzaam maar de tweede is heel erg snel (zie pag. 25 & 26 Engels handboek).

Regelnummers hebben we altijd wel genoeg (er kunnen er ruim 30.000 worden gebruikt) maar het aantal labels is beperkt tot 26. En als die op zijn moeten we onze toevlucht zoeken tot het gebruik van die langzame regelnummers. Er is echter een derde mogelijkheid om sprongen uit te laten voeren. Machinetaal- en diskdrive- gebruikers kennen allemaal wel het auto-startadres #CE86. De routine die op dit adres begint zet eerst het Y-register van de processor op nul en haalt dan de page-vector op, -deze staat op #12 (?18)-, en zet vervolgens de gevonden waarde in locatie #06. Bijbehorende locatie #05 wordt op nul gezet en evenzo de regel-karakterteller #03. Vervolgens wordt er gejumped naar #C55B.
Gesteld dat we een programma af hebben en dat alles goed werkt, maar dat verschillende routines wat langzaam zijn. Dit kan dan bijvoorbeeld komen door het veelvuldig gebruik van GOTO's naar regelnummers. Deze worden steeds langzamer naarmate de regelnummers hoger zijn. We nemen dan vooraan in het programma (regel 0 of zo) het volgende routinetje op:

```
0 P.$21;P=#....;[LDY@0;JMP#C55B;];P.$6
```
en op de plaats van .... zetten we het adres waar dit stukje machinetaal naartoe moet.
Nu staat bijvoorbeeld ergens in ons programma:
GOTO 1000
en de routine aldaar moet vaak worden gebruikt, dan kunnen we met b.h.v. 'n hexdump of met het verderop in deze uitgave beschreven WHATLINE opzoeken op welk adres regel 1000 begint. VOOR elk regelnummer staat een carriage-return (0D) en het is dan DAT adres dat we eventjes moeten onthouden.
Gesteld dat adres is: #34B7. I.p.v.:
GOTO 1000
vullen we dan in:
!5=#34B7;LINK.... en op de plaats van .... moet dan het adres komen waar het machinetaalroutinetje staat. Met name bij hogere regelnummers werkt deze methode vele malen sneller dan de gewone GOTO. En dit komt doordat de computer nu gebruik gaat maken van hetzelfde principe als bij het gebruik van labels:
Een adres wordt uit een tabel opgehaald en de basic-interpreter gaat dan verder met de uitvoering van de programmatuur op dat adres.
Een en ander zal misschien een aanzet zijn om iets uit te vissen zodat dit

truukje ook gebruikt mag worden voor GOSUB's.
LINK en GOTO mogen uiteraard worden afgekort hetgeen ook weer sneller werkt.

WHATLINE

```
 10DIMLL5;F.G=0TO5;LLG=-1;N.;P.$21;F.G=0TO1;P=#7000;[
 20LDA#33E;STA#81;LDY@0;STY#80;STY#28FF
 30:LL0LDA(#80),Y;INY;CMP@#D;BNELL0
 40TYA;LDY@0;CLC;ADC#80;STA#80;BCCLL1;INC#81
 50:LL1LDA(#80),Y;CMP@#FF;BNELL3
 60:LL2JSR#F7D1;];$P="EINDE PROGRAMMA";P=P+L.P;[NOP
 70JSR#CD54;RTS
 80:LL3LDA(#80),Y;CMP#33D;BNELL0;INY;LDA(#80),Y;CMP#322;BNELL0
 90LDA@1;STA#28FF;RTS;];N.;P.$6
100IN."BEGIN"B;@=5
110IN."REGEL"A;LI.#7000
120IF?#28FF=1P."REGEL "A" STAAT OP #"&?#81*256+?#80-1'
130G.110
```

Je kunt natuurlijk regel 110 vervangen door een lus zodat de adressen van ALLE regelnummers automatisch worden opgegeven.

?? (spreek uit: kweerie,kweerie)
Wat betekent ?#80=10. Op zero-pagelocatie #80 wordt dan de waarde 10 geplaatst.
M.a.w. letterlijk:
de inhoud van #80 wordt 10

Wat betekent !#80=#2A303406. Op zero-pagelocaties #80, #81, #82 en #83 komt dan te staan:
#80 = #06
#81 = #34
#82 = #30
#83 = #2A
Inderdaad precies in omgekeerde volgorde ingevuld.

Wat betekent G=#2800; G?#20=17.
G + #20 = #2820. Op locatie #2820 wordt dan 17 geplaatst.

In sommige programma's staan echter constructies als:
G??#80 of K!!F e.d.
Aan de hand van een voorbeeldje zal ik uitleggen wat hiermee wordt bedoeld:

```
?#80 = 6          inhoud van #80 wordt 6
G = #2800         inhoud van G wordt #2800
G?#10 = #80       inhoud van G + #10, dus
                  inhoud van #2810 wordt #80
G??#10 = 5        inhoud van G?#10 is #80
                  inhoud van #80 = 6
                  inhoud van G + 6, dus
                  inhoud van #2806 wordt 5
```

Lees bovenstaande regels aandachtig door en probeer ze ook eens uit. En kijk dan elke keer m.b.v. een hexdump naar het desbetreffende geheugengebied (hier dus #2800).
Het aantal ?'s en ook !'s mag zo groot zijn als men zelf wil. Houdt er wel rekening mee dat voor het lezen van contructies met 3 of meer ?'s of !'s heel wat gedachtensprongen nodig zijn. Een geoefend schaker of schaakster zal een en ander wel geinig vinden.

BOOLE

Denk niet dat dit wat met biljarten te maken heeft. De Atom kent een mogelijkheid om vergeijkingen te maken welke als resultaat kunnen hebben dat ze goed (TRUE) of fout (FALSE) zijn. Iets precieser: "WAAR" of "ONWAAR".
Probeer het volgende maar eens uit:

```
10 A=#2B00
20 B=#2C00
30 $A="BOOLE"
40 $B="BOOLE"
50 C=($A=$B)
60 IFC P."GELIJK"'; G.80
70 P."ONGELIJK"'
80 END
```

Na runnen van bovenstaand programmaatje zal er "GELIJK" op het scherm worden gezet.
Verander nu regel 30 in:

```
30 $A="POOLE"
```

Na runnen zal er "ONGELIJK" worden afgedrukt. C wordt nu gebruikt om een Boolse waarde in op te slaan. Als we voor alle twee de voorbeelden de waarde van C opvragen dan zal er de eerste keer komen te staan:

```
P.&C

   2B01
```

en de tweede keer:

```
P.&C

   2B00
```

Bij het stellen van de vraag:
IFC (de spatie hierachter is verplicht)
kijkt de computer alleen naar die laatste 1 of 0. Staat er 'n 1 dan geldt dat de vergelijking klopte, dus TRUE. Staat er 'n 0 dan geldt FALSE. Zodoende...
Het voordeel van deze methode is dat gekeken kan worden of iets klopt of niet en wel op zodanige manier dat plaats en tijd worden bespaard en met name plaats.
Dit soort geintjes kan natuurlijk ook voor andere zaken worden gebruikt:

K=((A*B+C*D)/(F*G)=12) of
T=(X=10)

maar we kunnen er ook in een keer een hele hoop dingen mee testen:

```
10 A=10;B=10;C=50;D=50;E=89;O=89
20 K=(((A=B)=(C=D))=(E=O))
30 IFK P."GELIJK"
40 END
```

zal na een run "GELIJK" op het scherm zetten. Vergelijk deze methode maar eens met:

```
10 A=10;B=10;C=50;D=50;E=89;O=89
20 IFA=B IFC=D IFE=O P."GELIJK"
30 END
```

Het zal dan onmiddelijk opvallen dat de eerste methode meer plaats gebruikt en dat er tevens een variabele extra moet worden gebruikt. Wanneer we echter herhaalde malen gebruik moeten maken van de vergelijking kunnen we in het eerste voorbeeld volstaan door elke keer te vragen:
IFK
IFK etc.

TELETABLIST

Programma ontwikkeld door J.v.ESSEN tbv. TELETYPE TOEPASSING.
Essentie is dat programma in 2 gelijke kolommen geprint wordt.

Link adres in regel 20 is teletype-output routine.

```
 10 REM TELETABLIST
 14 P.''
 15 P."VUL IN REGEL 20 ADRES"'
 16 P."VAN UW TELETYPE ROUTINE"''
 20 LINK#6A00;P.$2
 30 @=5;C=1;D=1
 35 P."GEEF HET STARTADRES VAN "'
 36 P."DE TE PRINTEN LISTING"'
 37 P."BV. #29     "''
 40 IN."STARTADRES/TEKSTSPACE"T
 50 T=T*256+3;S=T;R=0
 60 IFL.T>29;R=R+1;T=T+30;G.60
 70 R=R+1;T=T+L.T+3
 80 IFT?-2<>255G.60
 90 T=S;M=R/2;IFR%2;M=M+1
100 R=0
110 IFL.T>29;R=R+1;T=T+30;G.110
120 R=R+1;T=T+L.T+3
130 IFR<M;G.110
140 P=T;T=S;O=P
150 P.$15$2'''
160 E=C;Q=T;GOS.a;A=F;C=E
170 IFA=35;C=0;P."  ";G.210
180 IF?T=13;G.200
190 P.$(?T);A=A+1;T=T+1;G.170
200 T=T+3;F.I=A TO 36;P." ";N.
210 E=D;Q=P;GOS.a;B=F;D=E
220 IFB=35;D=0;P.';Q=P;GOS.c;G.160
230 IF?P=13;P=P+3;P.';Q=P;GOS.c;G.160
240 P.$(?P);B=B+1;P=P+1;G.220
250aIFE=0;P."        ";E=1;F=6;R.
260 GOS.c
270 G=(Q?-2)*256+(Q?-1);P.G;F=5
280 IF?Q<>32IF?Q<>58IF?Q<>91IF?Q<>93IF?Q<96P." ";F=F+1
290 R.
300cIF(Q?-2)=255IFO=T;G.b
310 R.
320bP.''';F.I=1TO20;P." ";N.;P."PROGRAMMALENGTE"Q-S+2" BYTES"''$3
340 ?18=?#80;E.
```

PIE-CHART VOOR GAGS

Dit programma genereert een zogenaamde pie-chart op basis van de invoergegevens. Gevraagd wordt een kopregel, hoeveel partjes en het aantal procent per partje en bijbehorende tekst. Probeert U zelf maar eens wat mogelijk is.
Dit programma heeft GAGS nodig plus geheugen tot in de 16K kaart op #4000.

```
  10 REM ********************
  20 REM *     PIE-CHART     *
  30 REM *        BY:        *
  40 REM *   THEO DEN EXTER  *
  50 REM ********************
  60
  70 REM **INPUT**
  80
  90 P.$12;GAGS;P.$12$15
 100 P."    ***PIE-CHART GENERATOR***"''''
 110 DIM W(64);P."HEADLINE: "
 120 DO;INP.$W;B=1
 130  FOR I=0 TO LENW;A=I?W
 140   IF A=CH"%" OR A=CH"." OR A=CH"-";G.v
 150   IF A<#30 OR A>=#5B;IF A<>#20 AND A<>#D;B=0
 160   IF #39<A AND A<=40;B=0
 170v NEXT
 180  IF B=0 OR LENW>20;P."NOT CORRECT, REENTER PLEASE:"'
 190  IF B=0 OR LENW>20;P."HEADLINE: "
 200 UNTIL LENW<21 AND B=1;P.''
 210 DO;INP."HOW MANY PIECES ",M
 220  IF M<2 OR M>8;P."NOT CORRECT, REENTER PLEASE:"'
 230 UNTIL 2<=M AND M<=8
 240 FDIM%MM(M);%R=0
 250 DIM TT(M),T(0);@=0;I=1
 260 DO;P.''
 270  IF I=M THEN %A=100-%R;G.d
 280  DO;P.I" - HOW MANY % ";FINP.%A
 290  UNTIL %A>=4
 300  FIF %R+%A<=96 THEN G.d
 310  FIF %R+%A>=100 THEN G.i
 320  %A=96-%R;G.d
 330i %A=100-%R
 340  P."NUMBER OF PIECES REDUCED TO "'I" UNITS"';M=I
 350d TT(I)=T
 360  DO;P."TEXT PIECE "I" ";INP.$T;B=1
 370   FOR J=0 TO LENT;A=J?T
 380    IF A=CH"%" OR A=CH"." OR A=CH"-";G.w.
 390    IF A<#30 OR A>=#5B;IF A<>#20 AND A<>#D;B=0
 400    IF #39<A AND A<=40;B=0
 410w  NEXT J
 420   IF B=0 OR LENT>10;P."NOT CORRECT, REENTER PLEASE:"'
 430  UNTIL LENT<11 AND B=1
 440  T=T+LENT+2
 450  %MM(I)=%A;I=I+1;%R=%R+%A
 460 UNTIL I>M
 470 P.'''"LAST PIECE "M" IS "
 480 P.%(%MM(M))" %"'''"PLEASE ENTER SPACE BAR"''
 490 LINK #FE94
 500
```

```
 510 REM **SET UP DISPLAY**
 520
 530 B=80;C=80;D=50
 540 BASE#98;CLEAR4;BORDER 5,5
 550 MOVE 250,162;DRAW 226,186
 560 CUBE 2,0,226,162,24,24
 570 CIRCLE 1,B,C,D
 580 CIRCLE 1,B,C,1;CIRCLE 1,B,C,2
 590
 600 REM **COLOUR PATTERNS**
 610
 620 CREATE/P:8 #55,#AA,#55,#AA,#55,#AA,#55,#AA
 630 CREATE/P:2 #44,#11,#44,#11,#44,#11,#44,#11
 640 CREATE/P:3 #99,0,#99,0,#99,0,#99,0
 650 CREATE/P:5 0,0,0,0,0,0,0,0
 660 CREATE/P:7 #66,#99,#99,#66,#66,#99,#99,#66
 670 CREATE/P:4 #22,#11,#88,#44,#22,#11,#88,#44
 680 CREATE/P:6 #44,#88,#11,#22,#44,#88,#11,#22
 690 CREATE/P:1 #FF,#00,#FF,#00,#FF,#00,#FF,#00
 700 CREATE/P:11 #88,#00,#00,#00,#88,#00,#00,#00
 710
 720 REM **DRAW**
 730
 740 GOS.b;REM **PLOT LINES**
 750 GOS.c;REM **COLOUR**
 760 PAUSE180
 770 PAINT 1,6;PAINT 254,96
 780
 790 DEF ACA,0000001111000000
 800 DEF:     0000011111100000
 810 DEF:     0000011111100000
 820 DEF:     0000111111110000
 830 DEF:     0000111111110000
 840 DEF:     0000111111110000
 850 DEF:     0000111111110000
 860 DEF:     0000000000000000
 870 DEF ACB,0001111111111000
 880 DEF:     0001111111111000
 890 DEF:     0000111111110000
 900 DEF:     0000001111000000
 910 DEF:     0001100110000000
 920 DEF:     0000011110000000
 930 DEF:     0000000111100000
 940 DEF:     0000000001111100
 950
 960 DEF COP,0001111000000000
 970 DEF:     0010000100000000
 980 DEF:     0101111010000000
 990 DEF:     0101000010000000
1000 DEF:     0101000010000000
1010 DEF:     0101111010000000
1020 DEF:     0010000100000000
1030 DEF:     0001111000000000
1040
```

```
1050 REM **HEADLINE**
1060
1070 K=20+7*LEN W
1080 X=(256-K)/2;IF LENW=0;X=30
1090 BLOCK0,X,159,K,19
1100 CUBE1,0,X,159,K,19
1110 SET: ACA,(X+3),176
1120 SET: ACB,(X+3),168
1130 IF LENW=0 THEN G.g
1140  FOR I=0 TO LEN W-1
1150   SET (W?I),(X+20+7*I),172
1160  NEXT
1170
1180  REM**COPYRIED**
1190g BLOCK 0,8,11,9,8
1200 SET:COP,8,16
1210 MOVE 5,5;PLOT 7,250,5
1220 PLOT 7,250,162;PLOT 7,226,186;PLOT 7,5,186
1230 PLOT 7,5,5
1240 CIRCLE 1,80,80,2
1250 PAUSE 100;INV
1260
1270 REM **WHAT TO DO NOW?**
1280
1290h G.h;REM **ROOM FOR SCREENDUMP**
1300
1310 REM--------------------
1320
1330a REM **MAIN PLOT LINE**
1340 X=%(B+7*SIN%A+0.5)
1350 Y=%(C+7*COS%A+0.5)
1360 MOVE B,C;DRAW X,Y
1370 E=%(X+(D+4)*SIN(%A)+0.5)
1380 F=%(Y+(D+4)*COS(%A)+0.5)
1390 PLOT 15,X,Y
1400 HLINE 1,X,Y,E,F,256,1,X,Y
1410 RETURN
1420
1430b REM **PLOT LINE**
1440 PAINT 6,6,11;PAINT 6,(C+D),11
1450 GOS.t
1460 %R=0;%A=0;GOS.a;%W=0
1470 FOR I=1 TO M
1480  IF I=M;G.r
1490  %R=%R+%MM(I)*2*PI/100
1500  %A=%R;GOS.a
1510r %K=%W+PI*%MM(I)/100;%W=%W+2*%MM(I)*PI/100
1520  X=%(B+(D+12)*SIN%K)
1530  Y=%(C+(D+12)*COS%K)
1540  X=X-8;Y=Y-4;BLOCK0,X,Y,18,12
1550  CUBE 1,0,X,Y,18,12
1560  K=%(%MM(I)+0.5);L=K%10+#30
1570  K=K/10+#30
1580  IF K=#30;SET:SPATIE,(X+3),(Y+9)
1590  IF K<>#30;SET K,(X+3),(Y+9)
1600  SET L,(X+8),(Y+9)
1610  SET:%,(X+13),(Y+9)
1620 NEXT I
1630 RETURN
```

```
1640
1650c REM**COLOUR**
1660 REM**COMPENSATE HLINE ERROR**
1670 REM**SHADE**
1680 CIRCLE 0,B,C,D
1690 CIRCLE 1,B,C,(D-1)
1700 WINDOW 1,(B+1),0,D,196
1710 CIRCLE 1,(B+1),C,(D-1)
1720 WINDOW 1,(B+2),0,(B+1),196
1730 CIRCLE 1,(B+2),C,(D-1)
1740 WOFF
1750 %R=0;A=1;%MM(0)=1
1760 FOR I=0 TO M-1
1770  %R=%R+(%MM(I))*2*PI/100
1780  X=%(B+(D-9)*SIN%R+0.5)
1790  Y=%(C+(D-9)*COS%R+0.5)
1800  PAINT X,Y,A
1810  %W=%R+(%MM(I+1)-1.5)*2*PI/100
1820  X=%(B+(D-9)*SIN%W+0.5)
1830  Y=%(C+(D-9)*COS%W+0.5)
1840  PAINT X,Y,A;A=A+1
1850 NEXT
1860
1870 REM **CUBES**
1880
1890 GOS.z;K=0
1900 FOR I=1 TO M;IF LEN TT(I)>0;K=K+1
1910 NEXT;IF K=0;G.p
1920 B=160;C=7+(K-1)*18
1930 I=0;S=0;DO;I=I+1
1940 IF LEN TT(I)=0;G.l
1950 S=S+1;K=K-1
1960  BLOCK0,B,(C-18*(S-1)),15,15
1970  CUBE 1,0,B,(C-18*(S-1)),15,15
1980  CUBE 1,0,(B+2),(C+2-18*(S-1)),11,11
1990  PAINT (B+3),(C+3-18*(S-1)),I
2000  Q=B+21;R=C-18*(S-1);T=256-11-Q;BLOCK 0,Q,R,T,15
2010  CUBE 1,0,(B+21),(C-18*(S-1)+1),(256-11-B-21),14
2020  GOS.y;REM **PLOT ALPHA**
2030lUNTIL K=0
2040pRETURN
2050
2060z REM **ALPHA**
2070 BASE#98
2080 CREATE ZERO,0,#70,#88,#C8,#A8,#98,#88,#70,0,0,0,0,0,0,0,0/A:CH"0"
2090 CREATE ONE,0,#70,#20,#20,#20,#20,#60,#20,0,0,0,0,0,0,0,0/A:CH"1"
2100 CREATE TWO,0,#F8,#80,#80,#70,#08,#88,#70,0,0,0,0,0,0,0,0/A:CH"2"
2110 CREATE THREE,0,#70,#88,#08,#30,#08,#88,#70,0,0,0,0,0,0,0,0/A:CH"3"
2120 CREATE FOUR,0,#10,#10,#F8,#90,#50,#30,#10,0,0,0,0,0,0,0,0/A:CH"4"
2130 CREATE FIVE,0,#70,#88,#08,#08,#F0,#80,#F8,0,0,0,0,0,0,0,0/A:CH"5"
2140 CREATE SIX,0,#70,#88,#88,#F0,#80,#40,#30,0,0,0,0,0,0,0,0/A:CH"6"
2150 CREATE SEVEN,0,#80,#80,#40,#20,#10,#08,#F8,0,0,0,0,0,0,0,0/A:CH"7"
2160 CREATE EIGHT,0,#70,#88,#88,#70,#88,#88,#70,0,0,0,0,0,0,0,0/A:CH"8"
2170 CREATE NINE,0,#60,#10,#08,#78,#88,#88,#70,0,0,0,0,0,0,0,0/A:CH"9"
2180 CREATE A,0,#88,#88,#88,#F8,#88,#88,#70,0,0,0,0,0,0,0,0/A:CH"A"
2190 CREATE B,0,#F0,#48,#48,#70,#48,#48,#F0,0,0,0,0,0,0,0,0/A:C "B"
```

```
2200 CREATE C,0,#70,#88,#80,#80,#80,#88,#70,0,0,0,0,0,0,0,0/A:CH"C"
2210 CREATE D,0,#F0,#48,#48,#48,#48,#48,#F0,0,0,0,0,0,0,0,0/A:CH"D"
2220 CREATE E,0,#F8,#80,#80,#F0,#80,#80,#F8,0,0,0,0,0,0,0,0/A:CH"E"
2230 CREATE F,0,#80,#80,#80,#F0,#80,#80,#F8,0,0,0,0,0,0,0,0/A:CH"F"
2240 CREATE G,0,#78,#88,#88,#88,#80,#88,#70,0,0,0,0,0,0,0,0/A:CH"G"
2250 CREATE H,0,#88,#88,#88,#F8,#88,#88,#88,0,0,0,0,0,0,0,0/A:CH"H"
2260 CREATE I,0,#70,#20,#20,#20,#20,#20,#70,0,0,0,0,0,0,0,0/A:CH"I"
2270 CREATE J,0,#70,#88,#08,#08,#08,#08,#08,0,0,0,0,0,0,0,0/A:CH"J"
2280 CREATE K,0,#88,#90,#A0,#C0,#A0,#90,#88,0,0,0,0,0,0,0,0/A:CH"K"
2290 CREATE L,0,#F8,#80,#80,#80,#80,#80,#80,0,0,0,0,0,0,0,0/A:CH"L"
2300 CREATE M,0,#88,#88,#88,#A8,#A8,#D8,#88,0,0,0,0,0,0,0,0/A:CH"M"
2310 CREATE N,0,#88,#88,#88,#98,#A8,#C8,#88,0,0,0,0,0,0,0,0/A:CH"N"
2320 CREATE O,0,#70,#88,#88,#88,#88,#88,#70,0,0,0,0,0,0,0,0/A:CH"O"
2330 CREATE P,0,#80,#80,#80,#F0,#88,#88,#F0,0,0,0,0,0,0,0,0/A:CH"P"
2340 CREATE Q,0,#68,#90,#A8,#88,#88,#88,#70,0,0,0,0,0,0,0,0/A:CH"Q"
2350 CREATE R,0,#88,#90,#A0,#F0,#88,#88,#F0,0,0,0,0,0,0,0,0/A:CH"R"
2360 CREATE S,0,#70,#88,#08,#70,#80,#88,#70,0,0,0,0,0,0,0,0/A:CH"S"
2370 CREATE T,0,#20,#20,#20,#20,#20,#20,#F8,0,0,0,0,0,0,0,0/A:CH"T"
2380 CREATE U,0,#70,#88,#88,#88,#88,#88,#88,0,0,0,0,0,0,0,0/A:CH"U"
2390 CREATE V,0,#20,#50,#88,#88,#88,#88,#88,0,0,0,0,0,0,0,0/A:CH"V"
2400 CREATE W,0,#88,#D8,#A8,#A8,#88,#88,#88,0,0,0,0,0,0,0,0/A:CH"W"
2410 CREATE X,0,#88,#88,#50,#20,#50,#88,#88,0,0,0,0,0,0,0,0/A:CH"X"
2420 CREATE Y,0,#20,#20,#20,#50,#88,#88,#88,0,0,0,0,0,0,0,0/A:CH"Y"
2430 CREATE Z,0,#F8,#80,#40,#20,#10,#08,#F8,0,0,0,0,0,0,0,0/A:CH"Z"
2440 CREATE SPATIE,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0/A:CH" "
2450 CREATE %,0,#88,#48,#40,#20,#10,#90,#88,0,0,0,0,0,0,0,0/A:CH"%"
2460 CREATE POINT,0,#30,#30,0,0,0,0,0,0,0,0,0,0,0,0/A:CH"."
2470 CREATE LINE,0,0,#F8,#F8,0,0,0,0,0,0,0,0,0,0,0/A:CH"-"
2480 RETURN
2490
2500y REM **TEXT**
2510 G=TT(I)
2520 IF LENK=0 THEN G.f
2530 FOR J=0 TO LEN G-1
2540  SET  (G?J),(B+24+J*6),(C+11-10*(S-1))
2550 NEXT
2560fRETURN
2570
2580t REM SMALL DIGITS
2590 CREATE 0,0,#40,#A0,#A0,#A0,#A0,#A0,#40,0,0,0,0,0,0,0,0/A:CH"0"
2600 CREATE 1,0,#40,#40,#40,#40,#40,#C0,#40,0,0,0,0,0,0,0,0/A:CH"1"
2610 CREATE 2,0,#E0,#80,#80,#40,#20,#A0,#40,0,0,0,0,0,0,0,0/A:CH"2"
2620 CREATE 3,0,#40,#A0,#20,#40,#20,#A0,#40,0,0,0,0,0,0,0,0/A:CH"3"
2630 CREATE 4,0,#20,#20,#E0,#A0,#80,#80,#80,0,0,0,0,0,0,0,0/A:CH"4"
2640 CREATE 5,0,#40,#A0,#20,#40,#80,#80,#E0,0,0,0,0,0,0,0,0/A:CH"5"
2650 CREATE 6,0,#40,#A0,#A0,#C0,#80,#80,#60,0,0,0,0,0,0,0,0/A:CH"6"
2660 CREATE 7,0,#80,#80,#80,#40,#20,#20,#E0,0,0,0,0,0,0,0,0/A:CH"7"
2670 CREATE 8,0,#40,#A0,#A0,#40,#A0,#A0,#40,0,0,0,0,0,0,0,0/A:CH"8"
2680 CREATE 9,0,#C0,#20,#20,#60,#A0,#A0,#40,0,0,0,0,0,0,0,0/A:CH"9"
2690 CREATE SPATIE,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0/A:CH" "
2700 CREATE %,0,#A0,#80,#80,#40,#20,#20,#A0,0,0,0,0,0,0,0,0
2710 RETURN
```

OPVOEREN VAN DE OLIVETTI DISK-DRIVE

De snelheid waarmee de kop van een disk-drive van spoor naar spoor beweegt is voor ieder merk verschillend en kan dan ook in de controller geprogrammeerd worden. Dit gebeurt bij het opstarten van de DOS dus met *D. of LINK#E000.
Op adres #E862 begint een tabel met initiatiewaarden: de derde waarde, 14, is de STEP-RATE. Dit komt overeen met 28 mSec. per stap, wat ook voor de Olivetti een overdreven grote waarde is.
Wanneer de drive op spoor 40 stilstaat en iets moet gaan lezen of schrijven, moet hij eerst 40 sporen heen en terug wat betekent dat je je meer dan 2 seconden zit te vervelen. Helemaal niet nodig want de drive werkt ook met een staptijd van 16 mS. en loopt dan op het gehoor zelfs nog soepeler. Het minimum is 12 mS. maar dat klinkt nogal ruw.
Het initieeren kan als volgt:

```
A=#A00 (commando register)
B=#A01 (parameter register)
?A=#35;?B=#D;?B=8;?B=10;?B=#EA
```

De eerste 2 waarden zijn de initiatiekode. De derde is de STEP-RATE (/2 zie boven). Nummer 4 is de HEAD SETTLING TIME, dit is de tijd die de kop krijgt om tot rust te komen nadat hij het doelspoor bereikt heeft en voordat het lezen of schrijven begint.
De vijfde waarde bestaat uit 2 parameters: de eerste 4 bits (hier #E) bepalen het aantal omwentelingen doorlooptijd na een opdracht. Het maximum is 14; 15 betekent dat de motor pas stopt na een andere initiatie of een break.
De tweede 4 bits is de HEAD LOAD TIME. HEAD SETTLING TIME en HEAD LOAD TIME zijn waarschijnlijk alleen van belang bij het formatteren. Eventueel kunnen grotere waarden dan hier gegeven geprobeerd worden als men problemen heeft met formatteren.
Niet initieeren terwijl de controller met een opdracht bezig is. Eigenlijk moet voor het schrijven gekeken worden of de registers al vrij zijn, maar Basic is lekker langzaam zodat dit geen problemen geeft.
De nieuwe initiatie moet uiteraard na de normale (*D.) uitgevoerd worden en blijft geldig na lees- of schrijfoperaties. Voor de kaart van Telec met de Teac drive geldt dit verhaal ook, alleen is de optimale STEP-RATE dan 3 (6 mS.). Deze waarde zit al in de standaardinitiatie.
Tot slot nog een assemblerversie van de initiatieroutine; deze vervangt *D.

```
 JSR#E000
 LDA@#35;JSR#E7DB
 LDA@#0D;JSR#E809
 LDA@#08;JSR#E809
 LDA@#0A;JSR#E809
 LDA@#3A;JSR#E809
 RTS
```

De eerste regel is de standaard initiatie subroutine. #E7DB is een routine die wacht met schrijven naar het commandoregister totdat het vrij is. #E809 idem voor het parameterregister.

GEWIJZIGDE FP ROM

Omdat me bekend is dat er nog al wat mensen zijn die zich ergeren aan de wat knullige presentatie van FP getallen, heb ik gezocht naar een mogelijkheid om binnen de bestaande FPROM tot wat meer vrijheden te komen.
Door een FP getal, namelijk %@, te laten vallen heb ik de volgende contruktie kunnen realiseren:
Als u in het code byte #2802 een nul invult dan wordt de inhoud van #2800 beschouwd als zijnde het aantal te printen decimalen (de punt meetellen). Door in het code byte een willekeurig getal in te vullen <> aan nul wordt teruggeschakeld naar de bekende FPROM, met dien verstande dat het punt waarop de E-notatie wordt ingeschakeld verlegd blijft van 0.5 naar plusminus 0.001. Als voorbeeld hier een klein programma dat deze zaak demonstreert.

```
10P.'"WERKEN MET NIEUWE FP ROM"'
20C=#2800;C?2=0;REM CODEBYTE OP NUL
30FDIM %GG5;DIMS12
40P.'"GEEF 5 FLOATING POINT GETALLEN"'
50FOR I=1 TO 5;IN.$S;%GGI=VALS;N.
60P.'"PRINTEN MET NIEUWE FP ROM"'
70%S=0;@=15
80?C=3; REM 2 DECIMALEN PRINTEN
90F.I=1 TO 5;FP.'%GGI;%S=%S+%GGI;N.
100@=8;FP.''"TOTAAL "%S''
110E.
```

```
RUN
WERKEN MET NIEUWE FP ROM
GEEF 5 FLOATING POINT GETALLEN
?0.01
?0.2
?4.65
?12.98
?19995.56

PRINTEN MET NIEUWE FP ROM

           0.01
           0.20
           4.65
          12.98
       19995.56

TOTAAL 20013.40
```

Probeer dit maar eens met de oude FPROM.
Tevens werkt dit afkappen ook met het STR commando.
bijvoorbeeld

```
W=#3800
STR PI,W
P.$W
```

geeft 3.14

WAARSCHUWING:

Doordat er wijzigingen zitten in het gebied #D400 tot #D593 kunnen er problemen optreden met BOXEN die hiervan gebruik maken.
Bij mij is dit tot nu toe alleen van de CALC-ROM bekend.
Deze springt naar #D547 (zowel JMP als JSR).
Dit moet gewijzigd worden in #D53D.
De twee plaatsen die in de CACL-ROM gewijzigd moeten worden zijn:

```
          #AB56    47 wordt 3D
          #ABA0    47 wordt 3D
```

En nu de wijzigingen in de FPROM.

```
D400 A9 00       LDA  @#00
D402 85 6F       STA #6F
D404 20 86 D6    JSR #D686
D407 D0 08       BNE #D4E1
D4D9 A9 30       LDA @#30
D4DB 20 8D D5    JSR #D58D
D4DE 4C 71 D5    JMP #D571
D4E1 10 05       BPL #D4EB
D4E3 A9 2D       LDA @#2D
D4E5 20 8D D5    JSR #D58D
D4E8 A9 00       LDA @#00
D4EA 85 6D       STA #6D
D4EC A5 59       LDA #59
D4EE C9 77       CMP @#77
D4F0 B0 08       BCS #D4FA
D4F2 20 A0 D6    JSR #D6A0
D4F5 C6 6D       DEC #6D
D4F7 4C EC D4    JMP #D4EC
D4FA C9 84       CMP @#84
D4FC 90 10       BCC #D50E
D4FE D0 06       BNE #D506
D500 A5 5A       LDA #5A
D502 C9 A0       CMP @#A0
D504 90 08       BCC #D50E
D506 20 1B D7    JSR #D71B
D509 E6 6D       INC #6D
D50B 4C EC D4    JMP #D4EC
D50E A5 59       LDA #59
D510 C9 84       CMP @#84
D512 B0 07       BCS #D51B
D514 20 D8 D6    JSR #D6D8
D517 E6 59       INC #59
D519 D0 F3       BNE #D50E
D51B 38          SEC
D51C A9 FF       LDA @#FF
D51E 20 36 D6    JSR #D636
D521 A5 5A       LDA #5A
D523 C9 A0       CMP @#A0
D525 B0 DF       BCS #D506
D527 A9 01       LDA @#01
D529 A4 6D       LDY #6D
D52B 30 0A       BMI #D537
D52D C0 08       CPY @#08
D52F B0 06       BCS #D537
D531 C8          INY
D532 A9 00       LDA @#00
D534 85 6D       STA #6D
D536 98          TYA
D537 85 70       STA #70
D539 A2 09       LDX @#09
D53B 86 54       STX #54
D53D 20 75 D5    JSR #D575
D540 C6 70       DEC #70
D542 D0 0F       BNE #D553
D544 A9 2E       LDA @#2E
D546 20 8D D5    JSR #D58D
D549 AD 02 28    LDA #2802
D54C D0 05       BNE #D553
D54E AD 00 28    LDA #2800
D551 85 54       STA #54
D553 C6 54       DEC #54
D555 D0 E6       BNE #D53D
D557 A5 6D       LDA #6D
D559 F0 16       BEQ #D571
D55B A9 45       LDA @#45
D55D 20 8D D5    JSR #D58D
D560 A5 6D       LDA #6D
D562 10 0A       BPL #D56E
D564 A9 2D       LDA @#2D
D566 20 8D D5    JSR #D58D
D569 38          SEC
D56A A9 00       LDA @#00
D56C E5 6D       SBC #6D
D56E 20 87 D5    JSR #D587
D571 A9 0D       LDA @#0D
D573 D0 18       BNE #D58D
D575 A5 5A       LDA #5A
D577 4A          LSR A
D578 4A          LSR A
D579 4A          LSR A
D57A 4A          LSR A
D57B 20 8B D5    JSR #D58B
D57E A5 5A       LDA #5A
D580 29 0F       AND @#0F
D582 85 5A       STA #5A
D584 4C 4E D6    JMP #D64E
D587 C9 0A       CMP @#0A
D589 B0 09       BCS #D594
D58B 09 30       ORA @#30
D58D A4 6F       LDY #6F
D58F 91 52       STA (#52),Y
D591 E6 6F       INC #6F
D593 60          RTS
```

VEEL SUCCES.

ACORN ATOM RHYTHM BOX

In het februari nummer van Elektuur stond een drumbox beschreven met interface voor een ZX-81.
Dat deze drumbox ook aan onze ATOM aan te sluiten is zal niemand verbazen,immers onze ATOM kan toch alles.
En van nu af aan kan die dus ook drummen .. jawel.
We sluiten de basis drumprint gewoon op de B-poort van onze VIA aan en wel als volgt: (zie ook fig.1)

punt:	is de:	naar:
BD	BASE DRUM	PIN 10 PL 6/7
CD	CONGA	PIN 9 PL 6/7
HB	HIGH BONGO	PIN 8 PL6/7
LB	LOW BONGO	PIN 7 PL6/7
LC	LONG CYMBAL	PIN 6 PL6/7
CL	CLAVES	PIN 5 PL6/7
MR	MARACAS	PIN 4 PL6/7
SD	SNARE DRUM	PIN 3 PL6/7

Deze verbinding maak je met een stukje bandkabel van ongeveer 25 cm.
Maak deze kabel dus niet te lang omdat de poort ongebufferd is bij langere kabels dus b.v. een 74LS244 er tussen plaatsen.
Vergeet niet om de aarde van de computer te verbinden met de drumprint.Een 9 aderige bandkabel zou voor deze verbinding dus voldoende zijn.

PROGRAMMA
De hierna komende listings zijn van de volgende programma's:

1 MENU-1
2 MENU-2
3 DRUM-S1

Run eerst de programma's MENU-1 en MENU-2 deze maken twee andere programma's aan n.1 DFACE1 en DFACE2 welke door het DRUM-source programma (3) worden gebruikt.
De bediening van het programma wordt d.m.v de MENU'S verduidenlijkt en behoeft geen verdere toelichting.
Het gehele gebeuren wordt in KLEUR op uw scherm getoverd.(als u natuurlijk over een kleuren ATOM beschikt.
Het hele gebeuren is geschreven om in toolkit op #Axxx te plaaten en zal dan vanaf #A100 worden geassembleerd.
Ook de twee MENU'S komen in deze pagina te staan van #AB00 tot #AFFF en worden door het hoofdprpogramma razendsnel naar uw scherm gecopieerd.
Er zal dus in een interpreter moeten worden toegevoegd plus een commando b.v DRUM om het geheel compleet te maken.
Rest mij om te melden dat u bij opstart in een soort rhythm box terecht komt waar u uit 8 verschillende ritme's kunt kiezen van ROCK tot SAMBA,in de tweede pagina waar u met de "N" van next terecht komt kunt u met de toetsen 1 t/m 8 uw eigen drum fantasie zijn beloop laten gaan.
Hierbij wil ik nog melden dat dit programma in een aantal ritme's nog voor verbetering vatbaar is maar toch al zeer drum "ECHT" overkomt,voor verbeteringen houd ik mij dus aanbevolen.
Dan wens ik u hierbij veel drum plezier enne ... DENK AAN DE BUREN

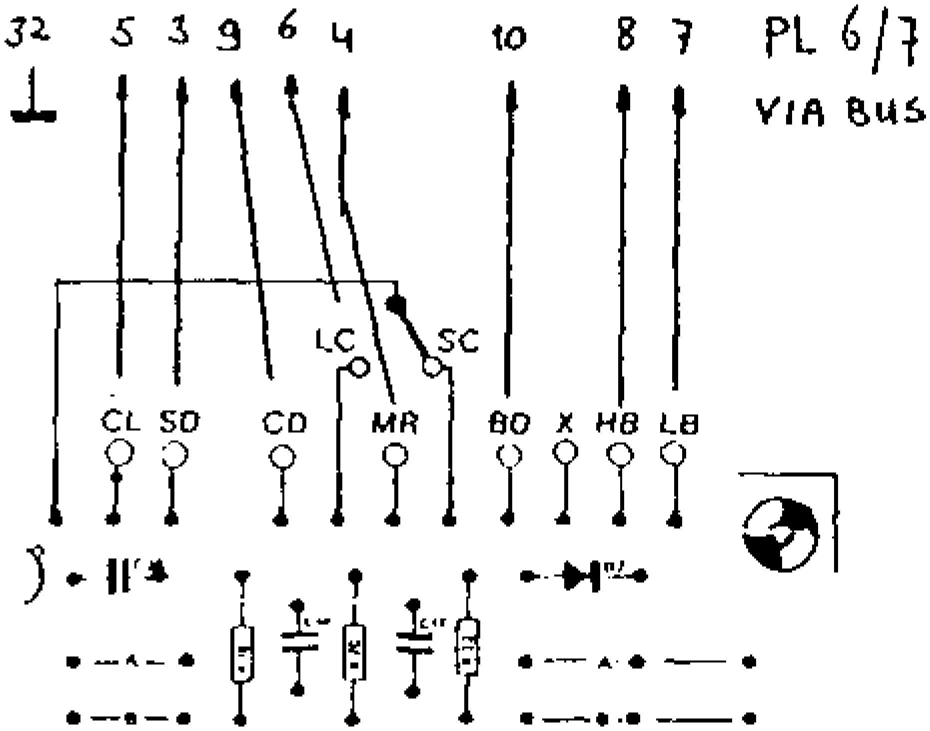


FIG. 1

```
 10REM MENU-1
 20REM MENU'S ZIJN IN KLEUR
 30P.$12;?#E1=0
 40?#B002 = 8
 50INV;REM INVERT SCREEN
 60F.I = 0TO33;P.$255;N.
 70P."   ACORN ATOM RHYTHM BOX  1 "
 80F.I = 0TO33;P.$255;N.
 90P."                                 "
100F.I = 0TO31;P.$255;N.
110P.$255"  1 = ROCK    "$234"  5 = SWING    "$255
120P.$255"  2 = WALTZ   "$234"  6 = MARCH    "$255
130P.$255"  3 = TANGO   "$234"  7 = DISCO    "$255
140P.$255"  4 = SAMBA   "$234"  8 = RUMBA    "$255
150F.I = 0TO31;P.$#FF;N.
160F.I = 0TO31;P.$#DF;N.
170P.$#DF"   CHOOSE A RHYTHM (1-8) ?    "$#DF
180P.$#DF"   CHOOSE A TEMPO  (<->) ?    "$#DF
190F.I = 0TO31;P.$#DF;N.
200P.$255$255"n"$157"next"$255$255$255"r"$157"restart"$255$255
210P.$255"s"
220P.$157"stop"$255$255$255
230?#81DD=160
240P.$21
250*S."DFACE1" 8000 81FF
260FOR I=0 TO 6000;NEXT
270P.$6;?#E1=#80
280END
```

```
 10REM MENU-2
 20REM MENU'S ZIJN IN KLEUR
 30P.$12;?#E1=0
 40?#B002 = 8
 50INV;REM INVERTEER HET SCHERM
 60F.I = 0TO33;P.$255;N.
 70P."   ACORN ATOM RHYTHM BOX  2 "
 80F.I = 0TO33;P.$255;N.
 90P."                              "
100F.I = 0TO31;P.$255;N.
110P.$255" 1 = BASS        "$234" 5 = BASS  * "$255
120P.$255" 2 = CONGA       "$234" 6 = SNARE * "$255
130P.$255" 3 = HIGH BONGO "$234" 7 = MARACAS "$255
140P.$255" 4 = LOW BONGO  "$234" 8 = SNARE   "$255
150F.I = 0TO31;P.$#FF;N.
160F.I = 0TO31;P.$#DF;N.
170P.$#DF"     CHOOSE A DRUM (1-8) ?    "$#DF
180P.$#DF"          status"$157"             "$#DF
190F.I = 0TO31;P.$#DF;N.
200P.$255$255$255$255$255$255$255$255"r"$157"restart"$255
210P."s"
220P.$157"stop"$255$255$255$255
230P.$255$255$255$255
240?#8188=160
250?#81D8=160;?#8189=160;?#8191=160;?#8192=160;?#8193=160
260?#8194=160;?#8195=160
270?#8196=160;?#8197=160;?#8198=160
280P.$21
290*S.DFACE2 8000 81FF
300FOR I=0 TO 6000;NEXT
310P.$6;?#E1=#80
320END
```

```
 10REM DRUM SOURCE VERSION 1
 20REM AREN SLOOTWEG
 30REM REGIO DEN HAAG
 40REM PROGRAMMA VOOR DRUMBOX
 50REM UIT ELEKTUUR FEB-1985
 60REM VOOR GEBRUIK IN TOOLKIT
 70REM WANNEER U INLINKT OP REGEL
 80REM 2090 JMP#C558 VERANDEREN IN RTS
 90P.$12''"DRUM source CODE"''
100*L.DFACE1 AD00
110*L.DFACE2 A800
120P.'"press A KEY"'"WHEN drive IS READY"''
130LINK#FFE3
140P."running"'
150DIMLL(100),SS(50)
160F.I=0TO100;LL(I)=-1;N.
170F.I=0TO50;SS(I)=-1;N.
180F.I=0TO1;P.$21
190P=#A100
200[
210:SS0
220\init
230LDA@27;STA#E0;LDA@#81;STA#DF;LDA@#80;STA#DE
240LDA@#FF;STA#B802
```

```
250LDA@0;STA#B800;TAY;TAX
260LDY@#40;STY#80\default wait
270:SS1LDA@8;STA#B002;LDY@0
280STY#E1
290:SS2
300LDA#AD00,Y;STA#8000,Y
310LDA#AE00,Y;STA#8100,Y
320INY;BNESS2;JMPLL18
330:LL15 \keyboardscan
340JSR#FE71;BCS LL16;JSR LL17
350:LL16
360STA#82;RTS
370:LL17
380PHP;CLD;JMP#FEB1
390:LL18
400JSRLL15
410LDA#82;CMP@#31;BNEP+5;JMPLL20
420LDA#82;CMP@#32;BNEP+5;JMPLL21
430LDA#82;CMP@#33;BNEP+5;JMPLL22
440LDA#82;CMP@#34;BNEP+5;JMPLL23
450LDA#82;CMP@#35;BNEP+5;JMPLL24
460LDA#82;CMP@#36;BNEP+5;JMPLL25
470LDA#82;CMP@#37;BNEP+5;JMPLL26
480LDA#82;CMP@#38;BNEP+5;JMPLL27
490LDA#82;CMP@#53;BNEP+5;JMPLL37
500LDA#82;CMP@#4E;BNEP+5;JMPSS10
510:LL31
520JMPLL18
530:LL0\base drum
540LDA@#01;STA#B800
550JSR LL50
560RTS
570:LL1\conga
580LDA@2;STA#B800
590JSRLL50
600RTS
610:LL2\high bongo
620LDA@4;STA#B800
630JSRLL50;RTS
640:LL3\low bongo
650LDA@#8;STA#B800
660JSRLL50;RTS
670:LL4\cymbale
680LDA@#10;STA#B800
690JSRLL50
700RTS
710:LL5\cymbal
720LDA@#20;STA#B800
730JSRLL50
740RTS
750:LL6\maracas
760LDA@#40;STA#B800
770JSRLL50
780RTS
790:LL7\snare drum
800LDA@#84;STA#B800
810JSRLL50
820RTS
```

```
 830:LL8\base&snare&chii
 840LDA@#85;STA#B800
 850;LDX@15;JSR#FB83
 860;LDA@#40;STA#B800;LDX@2;JSR#FB83;LDA@0;STA#B800;LDX@10
 870JSR#FB83
 880RTS
 890:LL9\base&cymbale
 900LDA@#11;STA#B800
 910JSRLL50
 920RTS
 930:LL10\base&maracas
 940LDA@#41;STA#B800
 950JSRLL50
 960RTS
 970:LL11\klap
 980LDA@#45;STA#B800
 990JSRLL50;RTS
1000:LL12\lowhighbongobaseconga
1010LDA@#0F;STA#B800
1020JSRLL50
1030:LL13\conga&maracas
1040LDA@#62;STA#B800
1050JSRLL50
1060RTS
1070:LL14\snare maracas
1080LDA@#C4;STA#B800
1090JSRLL50;RTS
1100:LL60\chi
1110LDA@#40;STA#B800
1120LDX@#5;JSR#FB83;LDA@0;STA#B800;JSRLL50;RTS
1130:LL61\base&snare
1140LDA@#85;STA#B800;JSRLL50
1150RTS
1160:LL62\roffel
1170:LL63 LDY@#FC;STY#86
1180:LL64 LDA@#85;STA#B800;LDX@2;JSR#FB83;LDA@0
1190STA#B800;LDX@1;JSR#FB83
1200INC#86;BNELL64;RTS
1210:LL66\trommel
1220:LL67 LDY@#F9;STY#86
1230:LL68 LDA@#85;STA#B800;JSRLL80;LDA@0
1240STA#B800;JSRLL90
1250INC#86;BNELL68;RTS
1260:LL50\wait loop
1270LDA@#F0;STA#81
1280LDA#85;STA#80
1290JSRLL35
1300:LL51 LDY@0
1310:LL53 INY;BNELL53;;INC#80;BNE LL53;
1320:LL54 LDY@0
1330:LL55 INY;BNELL55;INC#81;BNELL55;LDA@0;STA#B800
1340RTS
1350:LL80\wait loop klein
1360LDA#85;STA#80
1370:LL83 INC#80;BNE LL83;RTS
1380:LL90\wait loop middel
1390LDA#85;STA#80
```

```
1400JSRLL35;LDY@0
1410:LL91 INY;BNELL91
1420:LL92 INY;BNELL92
1430:LL93 INC#80;BNE LL92;RTS
1440:LL20\beat
1450LDA@#31;STA#817C
1460JSRLL61;JSRLL60;JSRLL13;JSRLL61;LDX@3;JSR#FB83;JSRLL61
1470JSRLL60
1480JSRLL13;JSRLL61;;LDX@3;JSR#FB83;JSRLL61;JSRLL60;JSRLL13
1490JSRLL61;LDX@3;JSR#FB83;JSRLL61;LDX@3;JSR#FB83
1500JSRLL61;JSRLL13;JSRLL3
1510JSRLL80
1520JMPLL20
1530:LL21\waltz
1540LDA@#32;STA#817C
1550JSRLL0;JSRLL6;JSRLL13;JSRLL0;JSRLL6;JSRLL6
1560JMPLL21
1570:LL22\tango
1580LDA@#33;STA#817C
1590JSRLL0;LDX@5;JSR#FB83
1600JSRLL0;LDX@5;JSR#FB83
1610JSRLL0;LDX@5;JSR#FB83
1620JSRLL0
1630JSRLL62;JMPLL22
1640:LL23\samba
1650LDA@#34;STA#817C
1660JSRLL0;JSRLL60;JSRLL60;JSRLL0;LDX@5;JSR#FB83;JSRLL0;JSRLL60
1670JSRLL60;JSRLL3;JSRLL80;;JMPLL23
1680:LL24\swing
1690LDA@#35;STA#817C
1700JSRLL13;JSRLL13;JSRLL13
1710JSRLL13;JSRLL13
1720JSRLL0;JMPLL24
1730:LL25\march
1740LDA@#36;STA#817C
1750JSRLL62;LDX@10;JSR#FB83
1760JSRLL10;JSRLL90
1770JSRLL10;JSRLL90
1780JSRLL62;LDX@10;JSR#FB83
1790JSRLL10;JSRLL90
1800JSRLL10;JSRLL90
1810JSRLL66;;LDX@10;JSR#FB83
1820JSRLL10;JSRLL90
1830JSRLL10;JSRLL90
1840 JMPLL25
1850:LL26\disco
1860LDA@#37;STA#817C
1870JSRLL0;JSRLL50;JSRLL7
1880JSRLL0;LDX@2;JSR#FB83
1890JSRLL0;JSRLL50
1900JSRLL7
1910JSRLL50;JMPLL26
1920:LL27\rumba
1930LDA@#38;STA#817C
1940JSRLL0;JSRLL50;JSRLL1;JSRLL2
1950JSRLL60;JSRLL50;JSRLL60;JSRLL50;JSRLL0;LDX@4;JSR#FB83;
1960JSRLL50;JSRLL0;JSRLL50
1970LDX@4;JSR#FB83;JSRLL10
1980JSRLL50
```

```
1990LDX@4;JSR#FB83;JSRLL10;LDX@4;JSR#FB83;;JSRLL50;JMPLL27
2000:LL35
2010JSRLL15
2020:LL32\sneller langzaam
2030LDA#82;CMP@#2E;BNELL33;CLC;LDA#80;ADC@#5;STA#80;STA#85
2040JSR#F802;LDA@#8;JSR#FFF4;LDA@#8;JSR#FFF4
2050RTS
2060:LL33
2070LDA#82;CMP@#2C;BNELL34;CLD;LDA#80;SBC@#5;STA#80;STA#85
2080JSR#F802;LDA@#8;JSR#FFF4;LDA@#8;JSR#FFF4
2090RTS
2100:LL34LDA#82;CMP@#52;BEQLL36;RTS
2110:LL36JMPSS0
2120:LL37JSR#FD1A;JSR#FD69;JMP#C558
2130:SS10\solodrums
2140LDY@0
2150:SS12
2160LDA#AB00,Y;STA#8000,Y
2170LDA#AC00,Y;STA#8100,Y
2180INY;BNESS12
2190:SS11 JSRLL15
2200LDA#82;CMP@#31;BNEP+5;JSRSS20
2210LDA#82;CMP@#32;BNEP+5;JSRSS21
2220LDA#82;CMP@#33;BNEP+5;JSRSS22
2230LDA#82;CMP@#34;BNEP+5;JSRSS23
2240LDA#82;CMP@#35;BNEP+5;JSRSS24
2250LDA#82;CMP@#36;BNEP+5;JSRSS25
2260LDA#82;CMP@#37;BNEP+5;JSRSS26
2270LDA#82;CMP@#38;BNEP+5;JSRSS27
2280LDA#82;CMP@#53;BNEP+5;JMPLL37
2290LDA#82;CMP@#52;BNEP+5;JMPSS0
2300JMPSS11
```

```
2310:SS20\base
2320LDA@#B1;STA#8192
2330LDA@1;STA#B800;LDX@5;JSR#FB83;LDA@0;STA#B800;RTS
2340:SS21\conga
2350LDA@#B2;STA#8192
2360LDA@2;STA#B800;LDX@10;JSR#FB83;LDA@0;STA#B800;RTS
2370:SS22\high bongo
2380LDA@#B3;STA#8192
2390LDA@4;STA#B800;LDX@10;JSR#FB83;LDA@0;STA#B800;RTS
2400:SS23\low bongo
2410LDA@#B4;STA#8192
2420LDA@8;STA#B800;LDX@10;JSR#FB83;LDA@0;STA#B800;RTS
2430:SS24\roffel base
2440LDA@#B5;STA#8192
2450:SS34 LDY@#F0;STY#86
2460:SS35 LDA@#01;STA#B800;LDX@2;JSR#FB83;LDA@0
2470STA#B800;LDX@2;JSR#FB83
2480INC#86;BNELL34;RTS
2490:SS25\roffel snare
2500LDA@#B6;STA#8192
2510:SS30 LDY@#FC;STY#86
2520:SS31 LDA@#86;STA#B800;LDX@2;JSR#FB83;LDA@0
2530STA#B800;LDX@2;JSR#FB83
2540INC#86;BNELL31;RTS
2550:SS26\maracas
2560LDA@#B7;STA#8192
2570LDA@#40;STA#B800;;LDX@5;JSR#FB83;LDA@0;STA#B800;LDX@2
2580JSR#FB83;RTS
2590:SS27\snare
2600LDA@#B8;STA#8192
2610LDA@#84;STA#B800;LDX@10;JSR#FB83;LDA@0;STA#B800;RTS
2620]
2630NEXT I;P.$6
2640END
```

SIGNAAL-ANALYSE EN REPRODUKTIE

DE A/D-CONVERTER.
fig. 1 - 1b.
Het omzetten van een analoog signaal (uit bv. een cassetterecorder) naar een door de computer herkenbaar digitaal signaal gebeurt in onderstaand schema door een Ferranti-IC, de ZN 427E-8. Het schema is van Rob van Dort uit Haarlo (zie ACORN NIEUWS april 1984) en is door mij op print gezet en uitgetest. Door het IC 4518 weg te laten en een jumper aan te brengen tussen pen 1 en pen 3 komt de klok van de ATOM ongedeeld op de schakeling te staan, en dit bleek de ZN nog aan te kunnen. De specificaties vermelden een mogelijke klokfrequentie van minimaal 900 kHz en 'typical' van 1 MHz.
Het IC is nu in staat om elke 9 microseconden een digitaal woord van 8 bits, afgeleid van een analoge spanning op de ingang van 0V tot 2.55V, op de uitgang te zetten. De spanningsdeler in het schema brengt de ingangsgevoeligheid op -5V tot +5V. Bij -5V wordt 0000 0000 en bij +5V 1111 1111 geproduceerd.
Een klokpuls van hoog naar laag start deze conversie en deze wordt betrokken van de CA2-poort van de VIA. Het digitale 8-bits woord wordt toegevoerd aan de A-poort van de VIA. Een eenvoudig machinetaal programma kan de opgehaalde DATA in achtereenvolgende plaatsen in het geheugen zetten.

```
:INIT;LDA@0;STA#B803    maak de A-poort ingang
      STA#80;TAY        maak het lage byte van de adrespointer
                        en het Y-register 0
      LDA@#38;STA#81    zet het hoge byte van de adrespointer
                        op pagina #38
      LDA@#A;STA#B80C   programmeer de VIA zo, dat er op de
                        CA2-poort een negatieve klok puls
                        komt bij elke lees/schrijfopdracht
:MAIN;LDA#B801;STA(#80),Y   lees de A-poort en plaats de DATA
                            op adres aangegeven door
                            de inhoud van geheugenplaatsen
                            #80 (lage byte) en #81 (hoge byte)
      INY;BNE MAIN          verhoog adres met 1 en test op
                            Y-waarde van #FF+1
      INC#81;LDA#81;CMP@#80 verhoog hoge byte adrespointer met 1
                            en test op eindadres geheugen
      BNE MAIN;JMP #C55B    indien bereikt dan terug naar BASIC
```

De belangrijkste cyclus is:

```
LDA#B801;STA(#80),Y;INY;BNE MAIN
   4          6        2     3  (getallen in microseconden)
```

Na elke leesopdracht (LDA#B801) geeft de CA2-poort een laagpuls af en start de A/D-converter met het samenstellen van een nieuw woord. De snelste conversie bedraagt dus 15 microseconden. Bij een schakeling met DMA (direct memory access) wordt geen machinetaal programma gebruikt maar wordt het beschikbare geheugen direct na het verhogen van de adrespointer met de DATA gevuld. Op die manier is met de ZN 427E-8 de maximale snelheid van 9 microseconden per meting te halen. Met 15 microseconden scoren we dan niet slecht. Elke seconde worden er nu zo'n 65.000 metingen verricht. Een geheugen van 16 kbyte is dan in 0.25 sec volgeschreven. Met een vertragingslus zijn er langere opneem-tijden mogelijk.

DE D/A-CONVERTER.
fig 2 - 2b.

Het omzetten van een digitaal- naar een analoog signaal gaat nog eenvoudiger (en goedkoper). Hiervoor gebruik ik de ZN 426E-8, eveneens een Ferranti-IC, dat in staat is binnen 1 microseconde een overeenkomstig analoog signaal te produceren tussen 0 V en 2.55 V.
Deze spanning wordt aan een instelbaar versterkertrapje toegevoerd en vervolgens aan een geluidsversterker.
De uit het geheugen opgehaalde DATA wordt nu aan de B-poort aangeboden. Door een wachtlus in het programma te zetten kan het signaal vertraagd of versneld weergegeven worden.

GRAFISCH WEERGEVEN.
Hoe het signaal opgeslagen wordt kan zichtbaar gemaakt worden in de hoogste grafische mode door hiervan een grafiek te laten tekenen. De Y-coordinaat wordt gelijk gemaakt aan de inhoud van geheugenplaatsen en er worden verbindingslijnen getrokken tussen de opvolgende punten. Het hier speciaal voor geschreven machinecode programma doet dit 5X sneller dan met BASIC PLOT opdrachten.

HET PROGRAMMA.
Het assembler gedeelte omvat 4 blokken:
....een 22 microseconden/byte A/D converter
....een D/A converter
....een 'snelle' plotter
....een 15 microseconden/byte ultra snelle A/D converter

Om de opneem- en weergavetijd te kunnen beinvloeden, is op regel 130 en 140 een vertragingslus ingebouwd. De waarde van het X-register maal 7 microseconde is dan de vertraging/byte. Deze waarde wordt vanuit het P-CHARME basic gedeelte aangepast in de procedure SPEED.

Op regel 740 kan een eigen grafische-dump routine gelinkt worden. (Hier JSR#1444)

DE TRANSISTOR TRACER
fig. 3 - 6
Naar een artikel uit Digital&Micro Electronics (aug/sep/1984).
I....De tor-tracer wordt bestuurd door de A-poort. Het uitgaande spanningssignaal wordt toegevoerd aan een digitale voltmeter die uitgelezen wordt door de B-poort. (zie A.N nov.1983, artikel van Rudi van Drunen). De A-poort wordt met stapjes van 1 verhoogd van 0 tot 256. Deze code wordt toegevoerd aan de D/A-converter (ZN 426E-8), die er een spanning van 0 t/m 2.55 Volt van maakt. Een OP-AMP versterkt deze spanning tot 0 -10.00 V, in stapjes van 0.04 V, welke toegevoerd wordt aan de Collector/Emitter van de test-tor, terwijl een constante stroom de Basis stuurt.
Deze Basis-stroom kan met 2 keuze-schakelaars op diverse niveaux ingesteld worden. De stroom kan met S3 ingesteld worden op resp. 5, 10, 15, 20 en 25 microampere. Met S2 wordt deze stroom vermenigvuldigd met een factor van resp. 0.5, 1, 2 of 5.
De Collector-stroom wordt voor Collector-spanningen van 0 t/m 10 V omgezet in spanningen van 0 tot maximaal 10 V.
De figuren 7a - 7d zijn voorbeelden van het programma analyse.

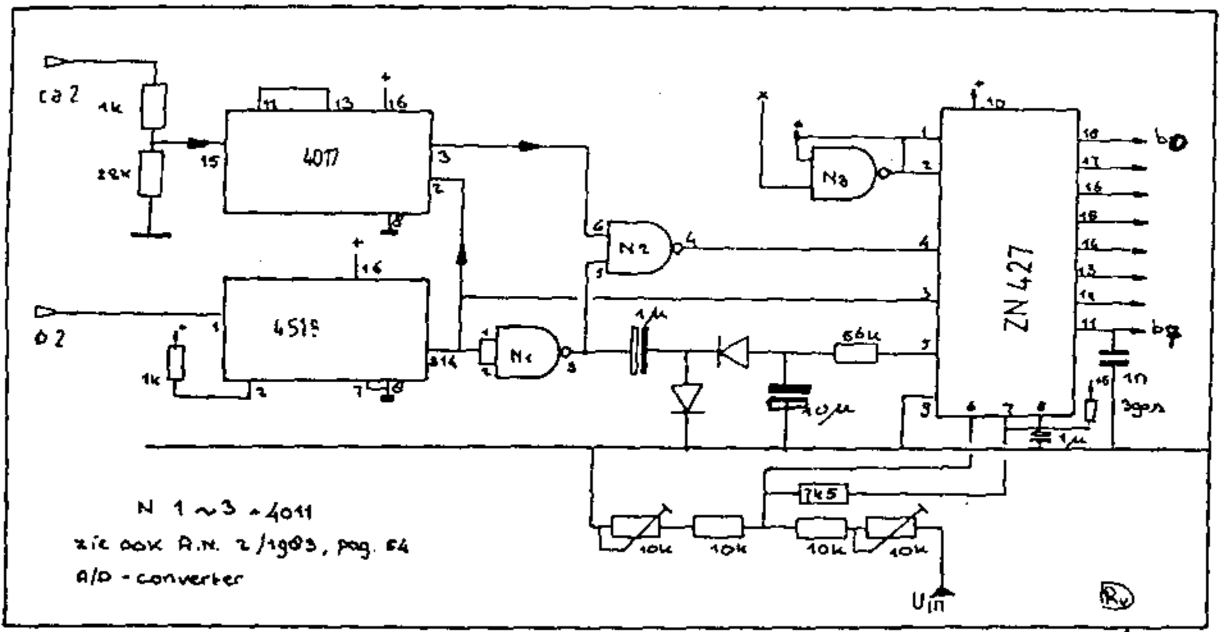
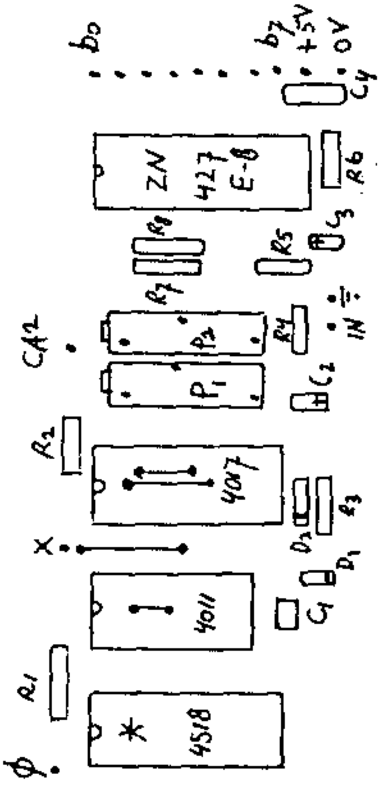
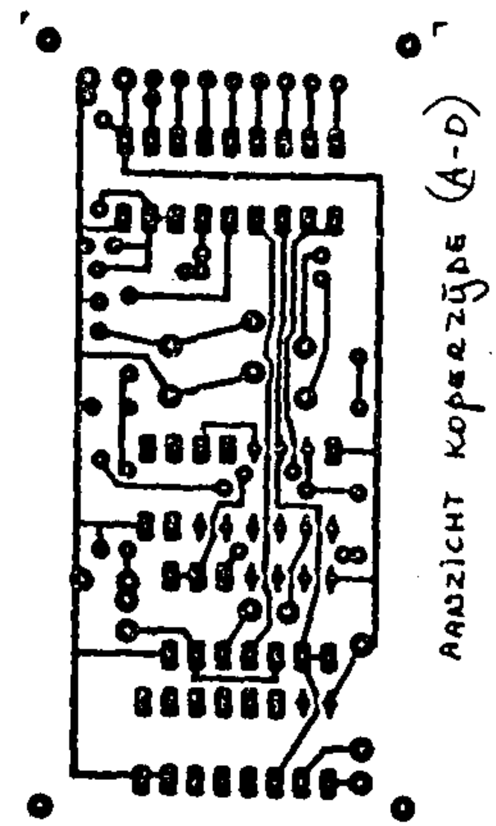

4017
ZN 427
N 1 ~ 3 = 4011
A/D - converter

fig 1.


ZN 427 E-8
4017
4011
4518


AANZICHT KOPERZIJDE (A-D)

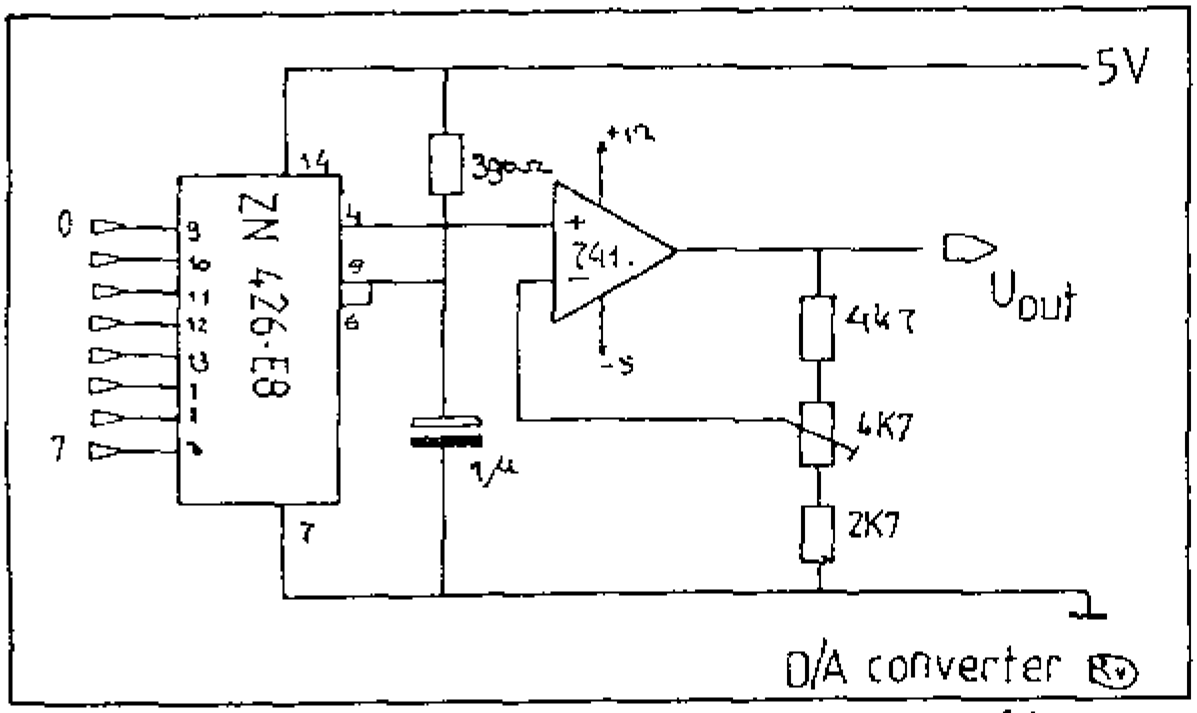
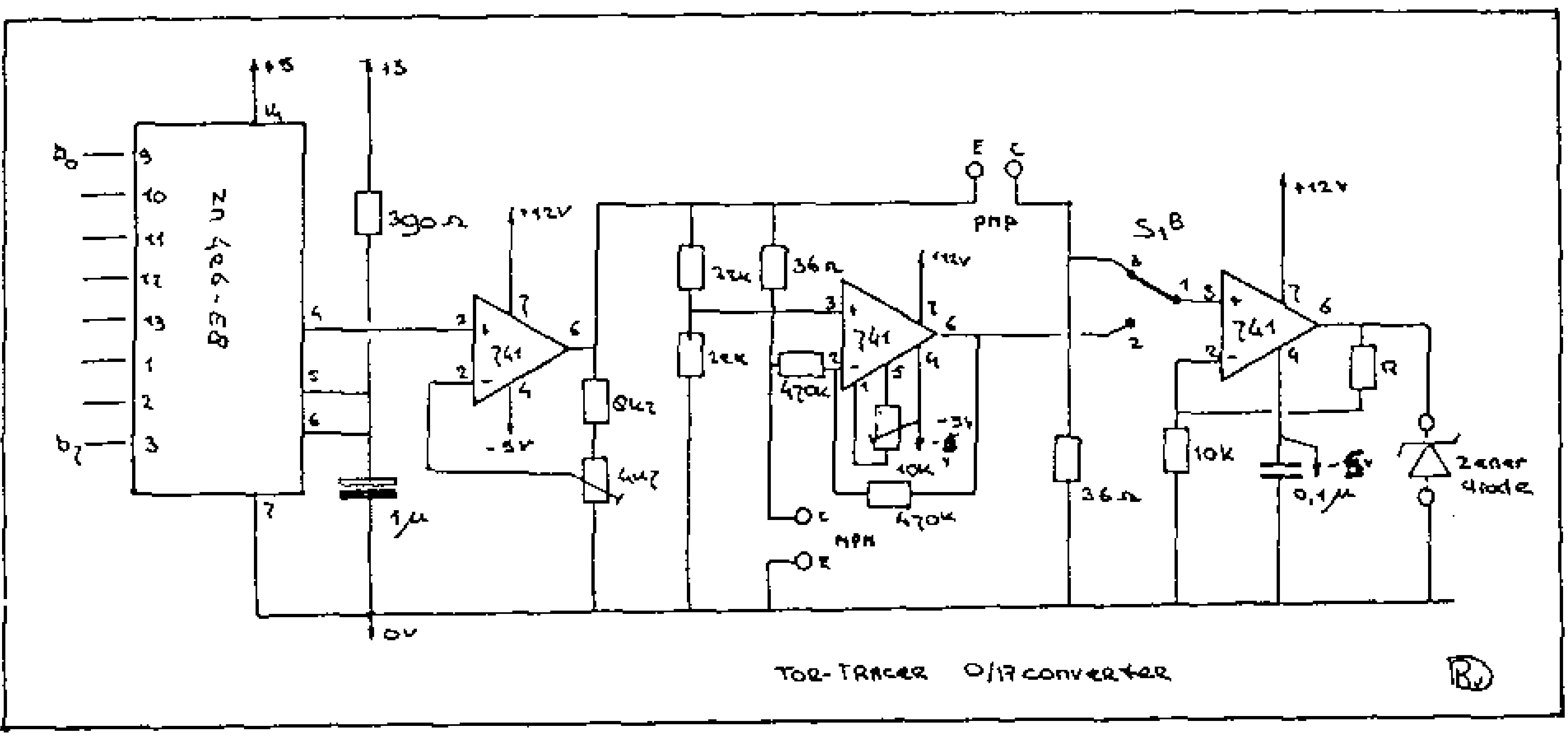

5V
+12
390Ω
ZN 426-E8
741.
-5
4K7
4K7
2K7
1μ
Uout
0
7
D/A converter

fig2.


+5
+5
ZN 426-E8
390Ω
+12V
741
-5V
4K7
4K7
1μ
0V
E
C
PNP
22k
36Ω
470K
10K
NPN
36Ω
S_1B
+12V
741
R
10K
0,1μ
zener diode
TOR-TRACER D/A converter

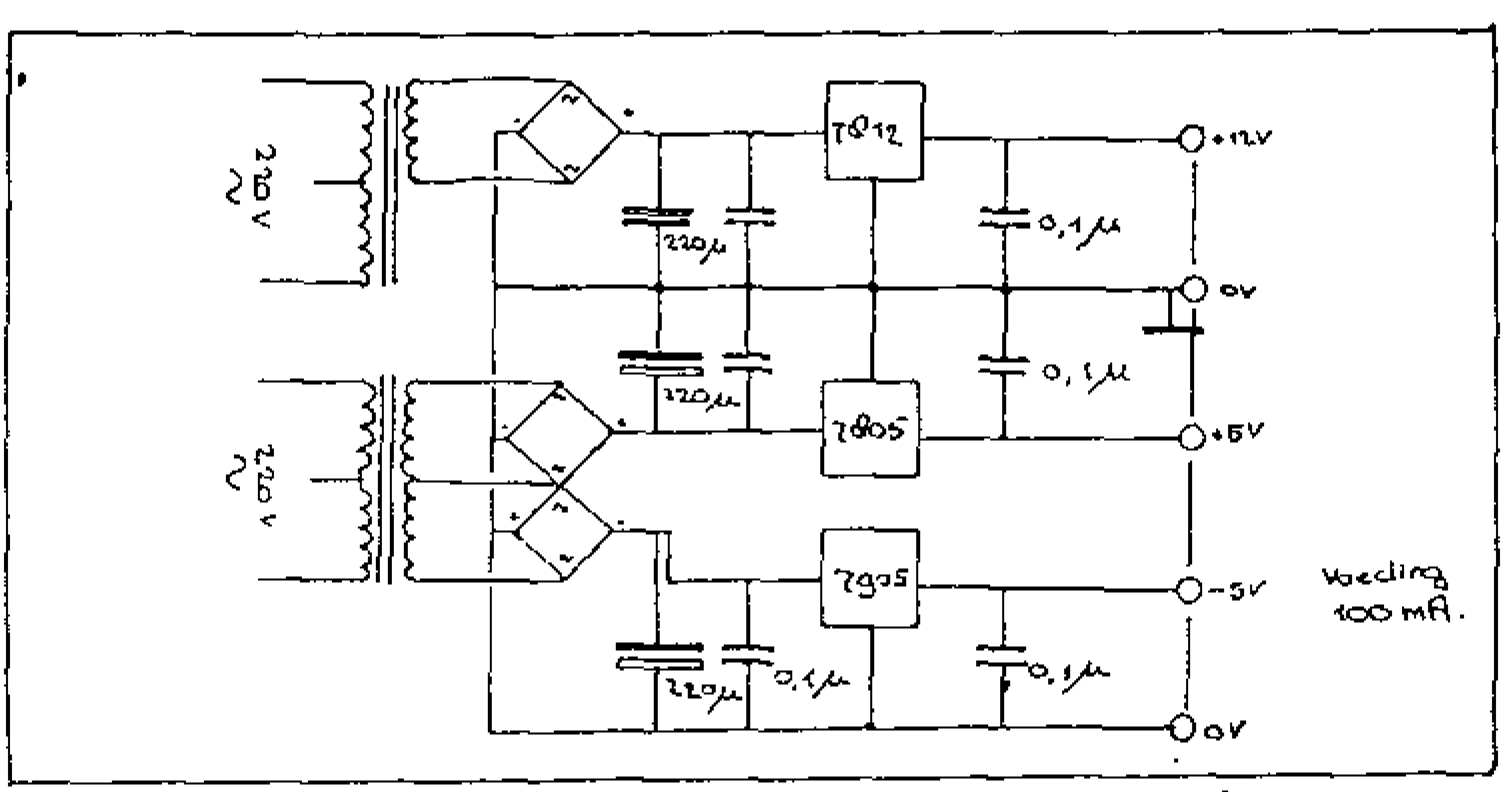
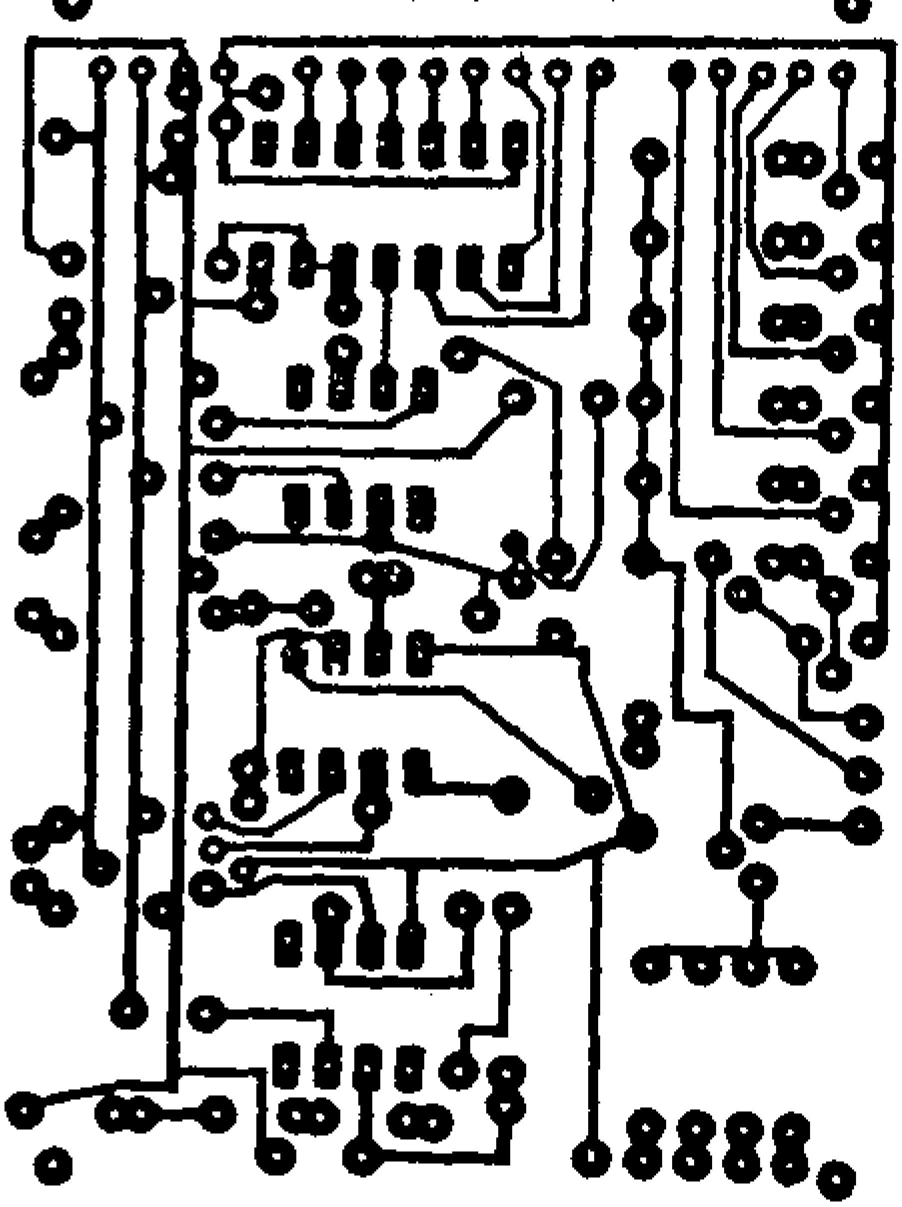
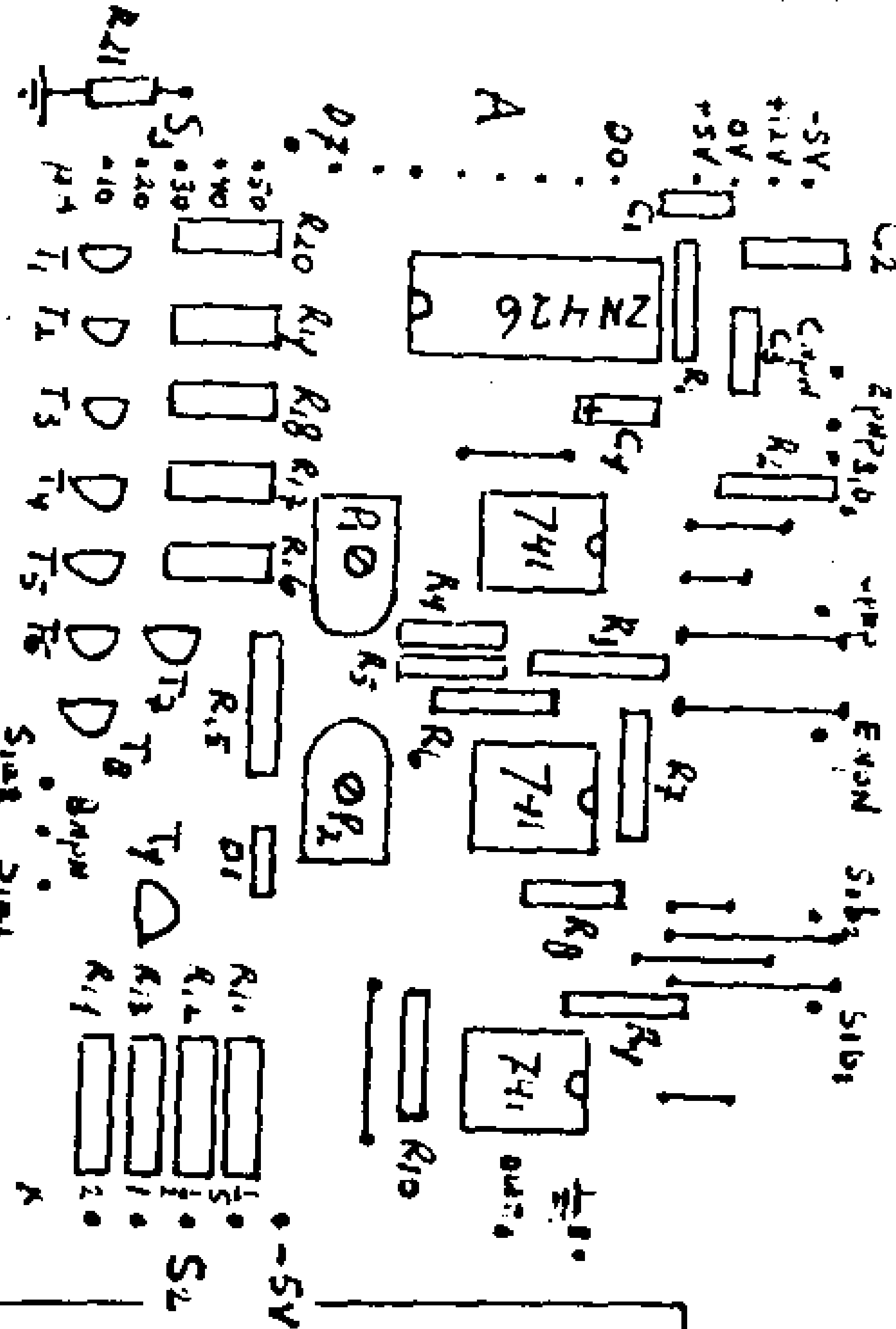

220V
220V
7812
7805
7905
220μ
220μ
220μ
0,1μ
0,1μ
0,1μ
0,1μ
+12V
0V
+5V
-5V
0V
voeding 100 mA.

fig.4.

TOR-TRACER





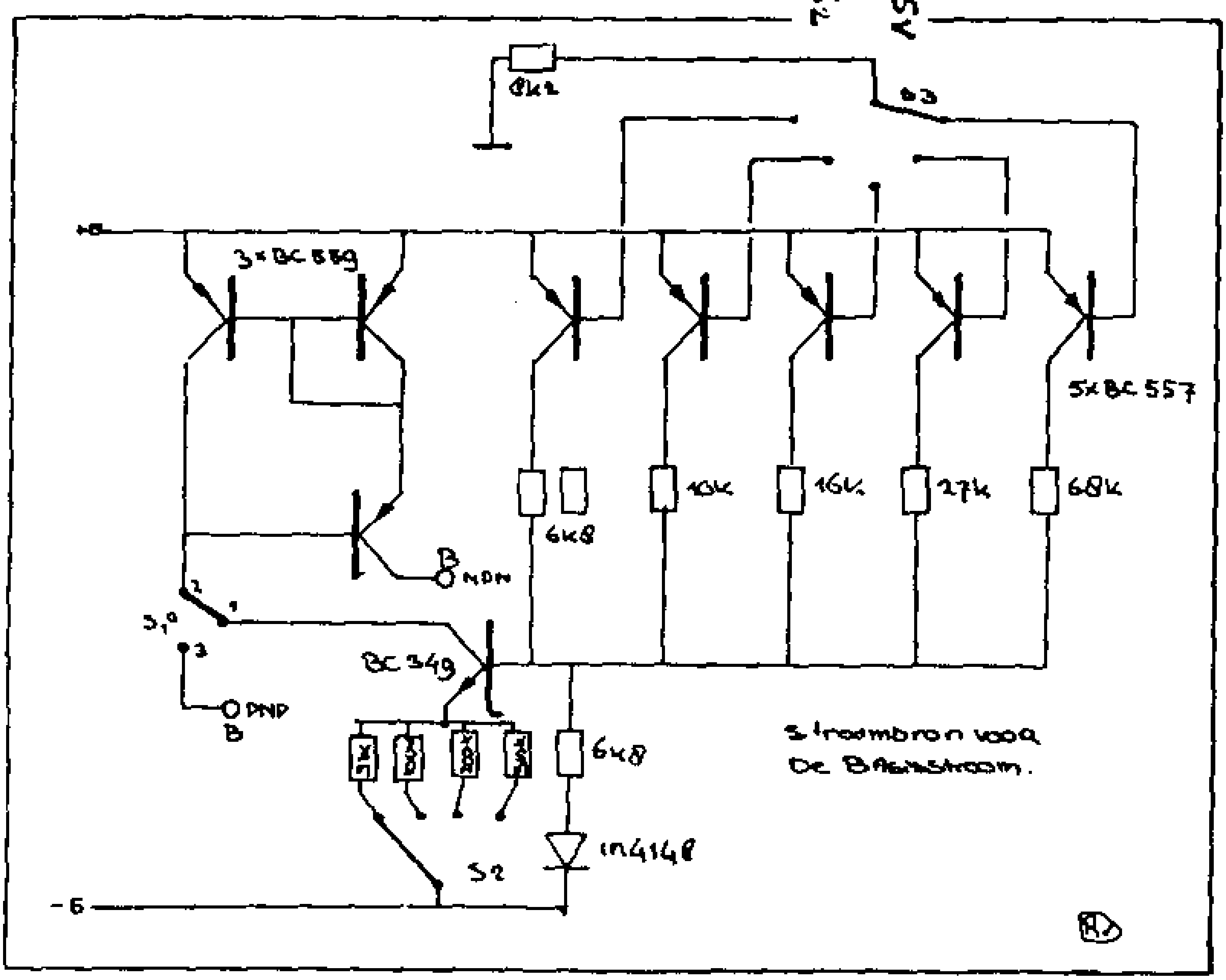
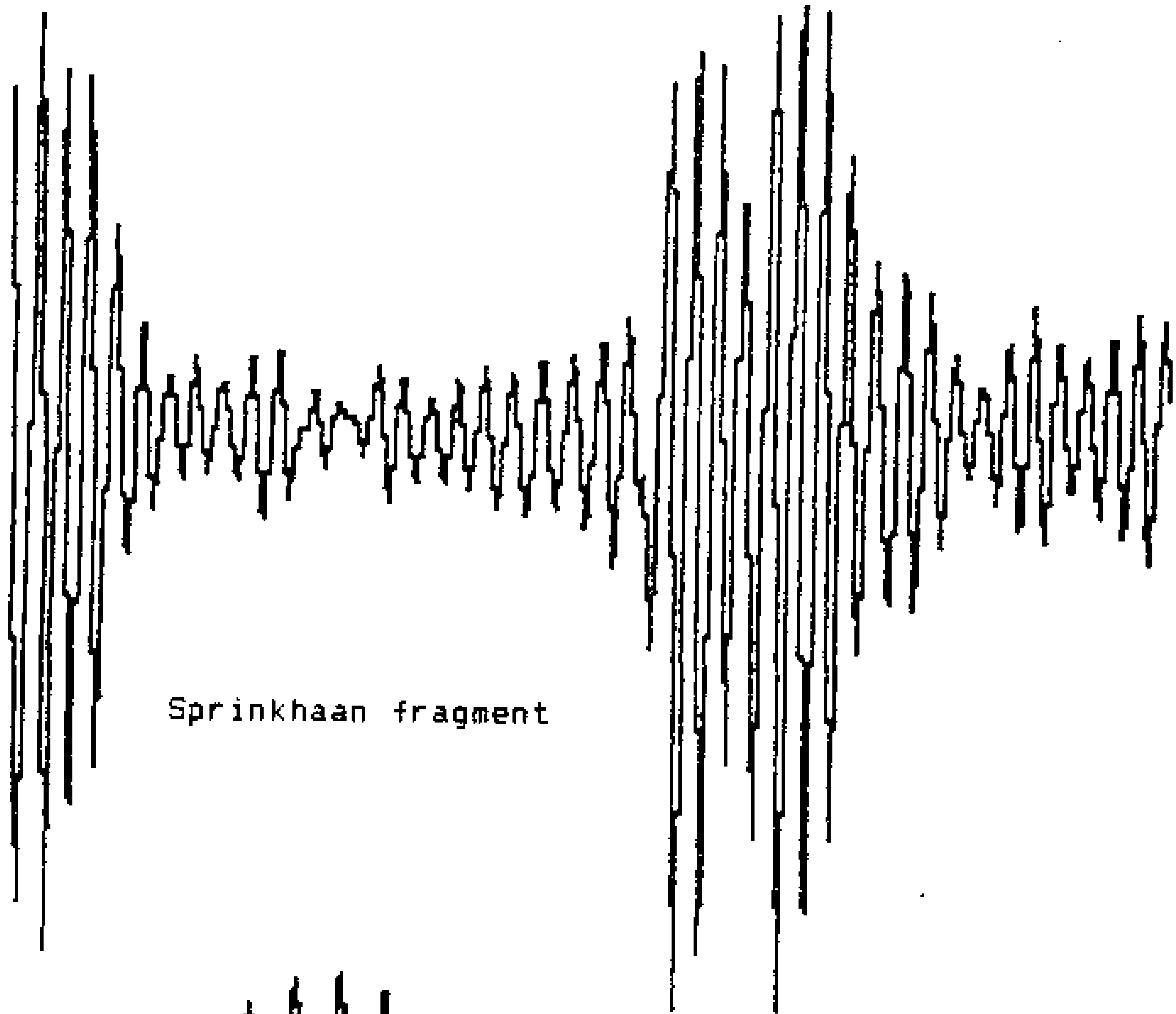
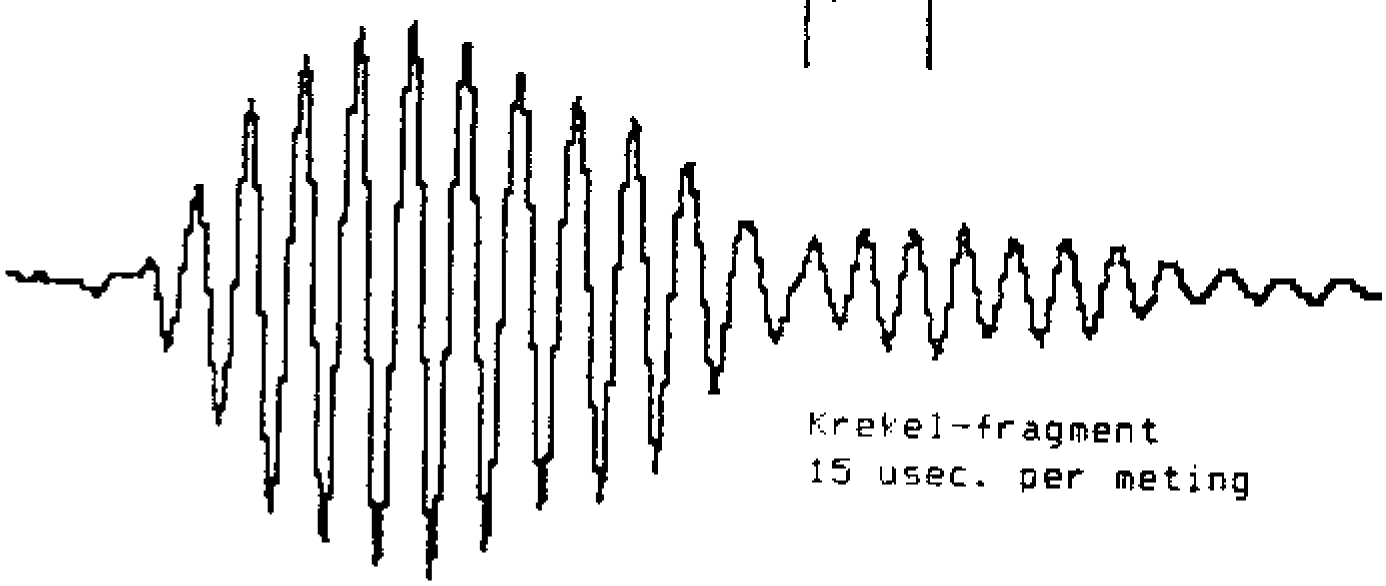


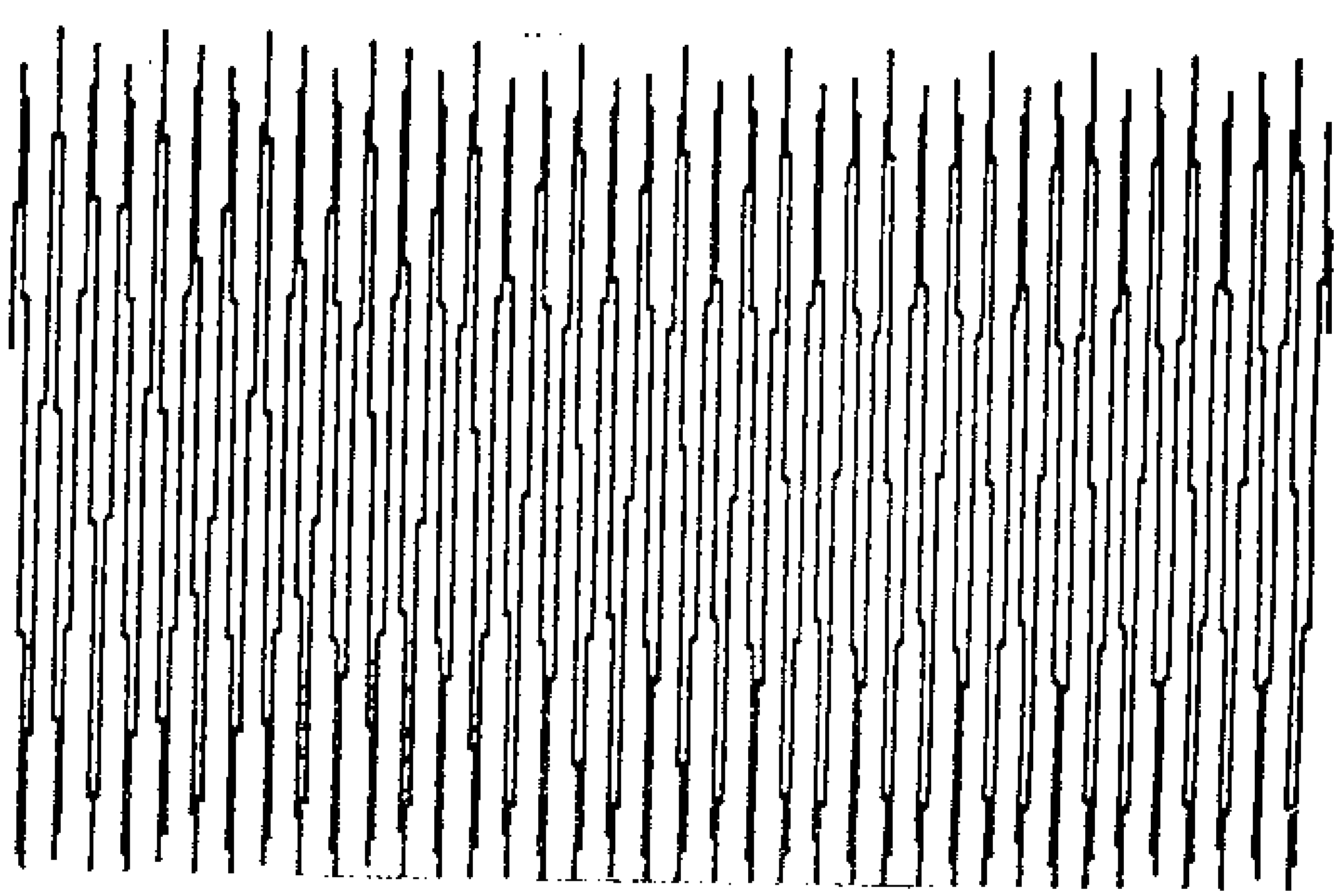
fig 5.



Sprinkhaan fragment



Krekel-fragment
15 usec. per meting



10.000 hz.

```
10 PROGRAM ANALYSE
20 REMARK zero-page:#80-#88
30 REMARK data-storage:#3800-#8000
40 DIMLL30;F.I=0TO30;LLI=-1;NEXT I
50 P.$21
60 F.I=0TO1;P=#3600
70[
80\22microsec a/d
90:LL0LDA@0;STA#B803;STA#80
100     TAY;LDA@#38;STA#81
110     LDA@#A;STA#B80C
120:LL1LDA#B801;STA(#80),Y
130     LDX@1
140:LL25DEX;BNELL25
150      INY;BNELL1
160      INC#81;LDA#81
170      CMP@#80;BNELL1
180      RTS
190\digital to analog
200:LL2LDA@#FF;STA#B802
210     LDA@0;STA#80
220     TAY;LDA@#38;STA#81
230:LL3LDA(#80),Y;STA#B800
240     LDX@1
250:LL26DEX;BNELL26
260      INY;BNELL3
270      INC#81;LDA#81
280      CMP@#80;BNELL3
290      RTS
300\fastplot
310:LL4LDA@#38;STA#81
320     LDY@0;STY#80
330:LL5LDA@#80;STA#88
340     LDA@0;STA#87;TAY
350:LL6STA(#87),Y;INY
360     BNELL6;INC#88
370     LDX#88;CPX@#98;BNELL6
380     LDA@1;STA#5E
390     LDA@#F0;STA#B000
400:LL7LDA(#80),Y;SBC@#20
410      STA#83;STA#5C;STY#5A
420:LL8INY;LDA(#80),Y
430      SBC@#20;STA#85
440:LL9ADC#83;BCSLL10
450     LSRA;STA#86;JMPLL11
460:LL10RORA;STA#86
470:LL11LDA#83;CMP#85;BCCLL15
480:LL12JSRLL22;CMP#86
490      BEQLL13;DEC#5C;JMPLL12
500:LL13INC#5A
510:LL14JSRLL22;CMP#85
520      BEQLL18;DEC#5C;JMPLL14
530:LL15JSRLL22;CMP#86;BEQLL16
540      INC#5C;JMPLL15
550:LL16INC#5A
560:LL17JSRLL22;CMP#85
570      BEQLL18;INC#5C;JMPLL17
580:LL18LDA#85;STA#83;STA#5C
590      CPY@#FF;BNELL8
600:LL19JSR#FFE3
610      CMP@#53;BEQLL23
620      CMP@#50;BEQLL21
630      CMP@#4E;BEQLL20
640      CMP@#47;BEQLL24
650      BNELL19
660:LL20INC#81;LDA#81
670      CMP@#80;BEQLL23;JMPLL5
680:LL21DEC#81;LDA#81
690      CMP@#37;BEQLL23;JMPLL5
700:LL22TYA;PHA;JSR#F7B0;PLA
710      TAY;LDA#5C;RTS
720:LL23JSR#FD1A;JMP#C55B
730:LL24JSR#1444;JMP#C55B
740\15microsec a/d
750:LL27LDA@0;STA#B803;STA#80
760      TAY;LDA@#38;STA#81
770      LDA@#A;STA#B80C
780:LL28LDA#B801;STA(#80),Y
790      INY;BNELL28
800      INC#81;LDA#81
810      CMP@#80;BNELL28
820      JMP#C55B
830];NEXT I;P.$6
840
850 PROC MENU
860     P.$12"      analoog-digitaal v.v."'
870     P."0...STOPPEN"'
880     P."1...INLEZEN ANALOOG SIGNAAL"'
890     P."2...GRAFISCHE WEERGAVE"'
900     P."     MET HARDCOPY MOGELIJKHEID"'
910     P."     (n)EXT     (p)REVIOUS"'
920     P."     (g)RAPHICS (s)TOP"'
930     P."3...WEERGAVE DIGITAAL SIGNAAL"'
940     P."4...INLEZEN 15 MICROSEC/BYTE"'
950     P.'"       keuze 0,1,. ..4"'
960 CHOICE
970 PEND
980
990 PROC CHOICE,K
1000     DO INKEY K
1010     UNTIL K>=CH"0" AND K<=CH"4"
1020     CASE K-CH"0" OF
1030<0>    END
1040<1>    !#208=#FE94FE55;SPEED(0)
1050<2>    LINKLL4
1060<3>    SPEED(2)
1070<4>    !#208=#FE94FE55;LINKLL27
1080     CEND
1090 PEND
1100
1110 PROC SPEED(S),A
1120 IF S=0 A=#3617
1130 IF S=2 A=#363B
1140 P.$12
1150     P."     vertraging"'
1160     P."1  2  4  8  16  32"'
1170     P."A  B  C  D   E   F"'
1180     DO INKEY K
```

```
1190     UNTIL K>=CH"A" AND K<=CH"F"
1200     CASE K-CH"A" OF
1210<0>     ?A=1;LINK LLS
1220<1>     ?A=4;LINK LLS
1230<2>     ?A=10;LINK LLS
1240<3>     ?A=23;LINK LLS
1250<4>     ?A=48;LINK LLS
1260<5>     ?A=98;LINK LLS
1270     CEND
1280 PEND
1290
1300 REMARK main
1310 DO
1320 MENU
1330 U.0
1340 END
```

onderdelenlijst TORTRACER:

```
R1     :390    C1,2,3:100N
R2,6   :36     C4    :1 MU
R3,5   :22K    T1, ,8:BC557
R4,21  :8K2    T9    :BC547
R7,8   :470K   D1    :1N4148
R9,17  :10K    P1    :4K7
R10    :40K*   P2    :10K
R11    :510K   IC'S  :ZN426E-8
R12    :200K          3X 741
R13    :100K   S1    :2X DUBBELOM
R14    :51K    S2    :4 STANDEN
R15,16:6K8     S3    :5 STANDEN
R18    :16K    BU1,2 :DIN5-180
R19    :27K
R20    :68K
```

R21 wordt los van de print opgenomen. Als de TORTRACER gebouwd wordt kan deze tegelijk als D/A converter voor de geluidsweergave van de opgenomen signalen gebruikt worden. Zet S1 dan in de stand PNP en zet een R van 15K tussen E en C van de PNP-bus. Afregeling: stuur de ZN volledig uit met 1111 1111 en regel P1 af op 10V op de E van de PNP-bus. Zet S1 op NPN en regel P2 af op 0V op de uitgang.

onderdelenlijst AD-CONVERTER:

```
R1,2   :1K     C1,3  :1 MU
R3     :22K    C2    :10 MU
R4     :56K    C4    :1N
R5     :7K5    D1,4  :1N4148
R6     :390    P1,2  :10K(10-SLAG)
R7,8   :10K    IC'S  :4518*,4011
                      4017,ZN427E-8
```

Pen 12 en 13 van de 4011 moeten

```
10 REM tor tracing program
20 REM DVM:B-POORT
30 REM D/A:A-POORT
40
50 DIM LL5
60 P.$21;F.I=0TO1;P=#2800
70[
80:LL2LDA@#80
90:LL1BIT#B800
100     BEQLL1;LDA@#60
110     BIT#B800;BNELL2
120     LDA#B800
130     AND@#0F;STA#90
140:LL4LDA@#40
150:LL3BIT#B800;BEQLL3
160     LDA@#A0;BIT#B800
170     BNELL4;LDA#B800
180     AND@#0F;STA#91
190:LL6LDA@#20
200:LL5BIT#B800
210     BEQLL5;LDA@#C0
220     BIT#B800;BNELL6
230     LDA#B800;AND@#0F
240     STA#92;RTS
250];N.;P.$6
260
270 REM graticule
280 CLEAR4
290 FOR I=20 TO 250 S.25
300   MOVEI,0
310   PLOT5,I,190
320 NEXT I
330 FOR J=20 TO 190 S.20
340   MOVE0,J;PLOT5,250,J
350 NEXT J
360 REM set-up
370 ?#B803=#FF;U=0;MOVE0,0
380 P.$7
390 LINK#FFE3
400
410 REM trace
420 FOR J=0 TO 4
430   FOR I=0 TO 256
440     ?#B801=I
450     LINK#2800
460     U=(?#90*100)+(?#91*10)+?#92
470     PLOT5,I,(U/5)
480   NEXT I
490   P.$7
500   MOVE0,0;LINK#FFE3
510 NEXT J
520 P.$7$7
530 LINK#FFE3;GOTO 270
```

P-CHARME SCHAKELSOFT DEEL 2

Met de komst van P-FUN (A.N. 3-7) voelde ik mij genoodzaakt de schakelsoft, eveneens in A.N. 3-7 gepubliceerd, aan te passen om ook gebruik te kunnen maken van de funktie-uitbreidingen. Het geheel is dermate kompakt geworden, dat zowel de schakelsoft als P-FUN tesamen slechts 252 bytes bedraagt en derhalve in een pagina past.
De schakelsoft werkt als volgt:
Allereerst wordt er getest of er een eprom gelockt staat (BIT#9FFF). Is dit niet het geval dan wordt P-Charme voorgeschakeld (in romvoet 7) en daarin wordt er gekeken of de te interpreteren opdracht afgehandeld kan worden. Is dit niet het geval dan springt de errorhandler van de P-Charme terug via een indirekte jump op #3EE naar adres #1100. Daar wordt altijd gesprongen naar label LL13, waar er wordt onderzocht welke fout is ontstaan. Er zijn nu drie mogelijkheden:

- ERROR 94: nu gaat de schakelsoft in werking treden: eerst worden de boxen afgezocht; wordt het statement daar niet gevonden dan wordt er vervolgens gesprongen naar een interpreter op #1102.
- ERROR 29: dan wordt er via P-FUN naar een funktie gezocht.
- indien er een andere error dan 94 of 29 gevonden wordt en als het statement of funktie niet gevonden is, wordt er naar de errorhandler van de P-Charme gesprongen.

De schakelsoft is wat sneller geworden, omdat de zeropage-adressen #90-#99 niet meer opgeslagen worden. Dit omdat de adressen #90-#97 alleen tijdelijk gebruikt worden door P-Charme en de adressen #98 en #99, gebruikt als datapointer voor READ en DATA, in de praktijk niet door andere statements gebruikt (hoeven) worden.
Om het geheel te initialiseren moeten de volgende opdrachten gedaan worden:

```
?#3EE=0    ?#3EF=#11    ?#3FB=#B3    ?#3FA=begin funktietabel
```

Deze opdrachten kunnen in een INIT-programma geplaatst worden of in de resetroutine.

Dient er nog opgemerkt te worden, dat alle funktie-uitbreidingen, ook die in P-Charme staan, niet in direkt mode gebruikt kunnen worden. De reden hiervan is, dat tijdens een BASIC-programma, dat P-Charme gebruikt, de breakvektoren (#202,#203) veranderd worden. In direkt mode wordt iedere keer als er een <return> gegeven wordt door de BASIC-rom de breakvektor op default (#C9D8) gezet. Het is dus niet mogelijk om zonder meer in direkt mode error 29 af te vangen. Wel kan er in de F.P.-rom het een en ander veranderd worden, maar daarover een andere keer.

```
 10 REM Schakelsoft P-CHARME en P-FUN
 20 REM versie 2.0
 30 REM basisadres:  #1000;
 40 REM schakelbyte: #9FFF
 50 REM uitbreiding: #1102
 60 Q=#1000;DIM LL18
 70 PRINT $21
 80 FOR I=Q TO Q+#FC STEP 4;!I=0;NEXT
 90 FOR I=0TO18;LLI=0;NEXT
100 FOR I=0TO1;P=Q+2;[
110:LL0
120                             \start
130 BIT#9FFF;BVS LL9
140                             \kijk eerst in p-charme
150 LDA@#AB;STA#203
160 LDA@#D5;STA#202
170 JSR LL12;BNE LL2
180:LL1
190 STA#BFFF
200 LDA#A000;CMP@#40;BNELL3
210 LDA#A001;CMP@#BF;BNELL3
220:LL2
230 JMP#A002
240:LL3
250 JMP#C558
260:LL4
270                             \ga schakelen
280 LDA@LL7%256;STA#202
290 LDA@LL7/256;STA#203
300:LL5
310 LDX@#FF;TXS
320 DEC#9FFF;LDA#9FFF
330 BPL LL1
340 JSR LL12
350 LDA@#4F;STA#202
360 LDA@#AC;STA#203
370:LL6
380                             \spring naar #1102
390 JMP #1102
400:LL7
410                             \error bij schakelen
420 PLA;PLA;STA0
430 CMP@#5E;BEQLL5
440 JSR LL12
450                             \terug naar p-charme
460:LL8
470 JMP#AC4F
480:LL9
490                             \gelockte eprom
500 LDA@LL10%256;STA#202
510 LDA@LL10/256;STA#203
520 LDA#9FFF;BNE LL1
530:LL10
```

```
 540                                \error bij gelockte eprom
 550 PLA;PLA;STA0
 560 CMP@#5E;BNELL11
 570 LDA@#C9;STA#203
 580 LDA@#D8;STA#202;BNELL6
 590:LL11
 600 JMP#C9D8
 610                                \subroutines
 620:LL12
 630 LDA@7;STA#9FFF;STA#BFFF;RTS
 640:LL13
 650                                \error p-charme:JMP(#3EE)
 660 PLA;PLA;STA0
 670 CMP@#5E;BEQLL4
 680 CMP@#1D;BNE LL8
 690                                \error 29 (start p-fun)
 700:LL14
 710 TAY;PLA;STA#90;LDX@#00
 720:LL15
 730 LDA#91,X;PHA;INX
 740 CPX@#07;BNE LL15;STY#97
 750 LDY@#03;STY#93;LDA@#A4
 760 STA#94;JSR#A8EA
 770 JSR#A52B;BCS LL16
 780                                \functioncall
 790 JMP#AB9C
 800                                \extra functions
 810:LL16
 820 LDA#3FA;STA#92
 830 JSR LL17;LDY@#03;JMP#AB82
 840:LL17
 850 LDY@#00;STY#91
 860 LDA#92;STA#94
 870 LDA(#91),Y;STA#92;INY
 880 STY#93;DEY;LDA(#93),Y
 890 CMP@#E3;BNE LL18;INY
 900 LDA(#93),Y;CMP@#C7
 910 BNE LL18;JSR#A8EA;JMP LL17
 920                                \not found
 930:LL18
 940 LDY#97;STY#00
 950 JSR#ADC6;INX;INX;TXS
 960 JMP#AC60
 970];NEXT
 980 @=0;R=P
 990 ?Q=#40;?(Q+1)=#BF
1000 P=Q+#100;[;BEQ LL13;]
1010 PRINT $6'"SCHAKELSOFTROUTINE: #"&Q+2"-#"&R';END

1020 in initialisatieprogramma het volgende opnemen:
1030 ?#3EE=0;?#3EF=#11
1040 ?#3FB=#B3
1050 ?#3FA=begin funktietabel
```

ERROR 29

Er werd altijd op onze regio-avonden gesproken, dat alleen ERROR 94 door verandering in floating-point rom afgevangen kon worden. Daardoor kunnen we door de aangebrachte driewegwissel in de FP-rom via schakelsoft naar statements gaan zoeken.
Maar ook ERROR 29 kan op een soortgelijke wijze afgevangen worden. Als bewijs de onderstaande STEP-routine met een nietbestaande funktie ONZIN:

```
>STEP#C2CF,#CA24
>A=ONZIN
CA24:JSR #C424   4F 01 03 21 F5
C424:LDA #D000   AA 01 03 A1 F5
C427:CMP @#AA    AA 01 03 23 F5
C429:BNE #C463   AA 01 03 23 F5
C42B:LSR         55 01 03 20 F5
C42C:CMP #D001   55 01 03 23 F5
C42F:BNE #C463   55 01 03 23 F5
C431:LDY #5E     55 01 02 21 F5
C433:RTS         55 01 02 21 F7
CA27:BCC #CA1B   55 01 02 21 F7
CA29:JMP (#D004) 55 01 02 21 F7
D15E:STY #03     55 01 02 21 F7
D160:JSR #D0EC   55 01 02 21 F5
>(escape)
```

Nu wordt er in de tabel van de FP-rom gezocht en als er niets gevonden wordt, gebeurt er het volgende:

```
>STEP#C2CF,#D296
>A=ONZIN
D296:BIT #73     96 01 02 E0 F5
D298:BMI #D2C0   96 01 02 E0 F5
D2C0:JMP #CA1B   96 01 02 E0 F5
CA1B:BRK
ERROR 29
>
```

Als de JMP #CA1B op adres #D2C0 nu veranderd wordt in een jump naar een adres, waar een routine staat, die funktie-uitbreidingen afloopt (bijv. P-FUN), dan is het ook mogelijk om de extra funkties in direkt mode te gebruiken. Bovendien wordt de BASIC er zeker niet trager door.

SCREENROM

De auteur is A.E.Knowles, 15 Belgrave Ave, Flixton, Manchester.
Deze rom is gebaseerd op een programma van computer consept.

Hieronder volgt een vrije vertaling van de gebruiksaanwijzing
die bij deze rom onmisbaar is.

Inleiding

Deze rom biedt de navolgende mogelijkheden.
- 32 of 42 tekens per regel met 24 regels op het scherm,
- witte of zwarte letters,
- tekstwindows, binnen de schermafmetingen,
- in elke mode (CLEAR 0-4) is tekst mogelijk (in "lagere mode dan 4 uiteraard met "grotere" letters.),
- u kunt zonodig uw eigen letterset maken (default is een standaard set upper- en lowercase (grote- en kleine-) opgenomen),
- auto repeat op alle toetsen,
- load en save (300 en 1200 baud) met zichtbare indicatie,
- uitvoer naar het scherm te onderbreken met CTRL+SPACE,
- het commando COLD (LINK#AFF0) "verplaatst" de standaard letterset naar #2900 tot #2BFF en verzet de tekstruimte pointer naar #2C00, wist het scherm en selecteert graphicsmode 4,
- het commando WARM (LINK#AFF3) werkt nagenoeg hetzelfde de letterset op #2900 blijft echter ongewijzichd,
- #80 tot #8A worden gebruikt, (conflicten niet uitgesloten).

Het cassette operating gebeuren

Met SLOW en FAST (LINK#AFF9 c.q.#AFF6) wordt gekozen tussen 300 en 1200 baud. De rom behoeft voor het gebruik van deze routines niet te worden geinitialiseerd. In de rechter bovenhoek van het scherm is het verloop van het proces te volgen. Bij het beeindigen hoort u een paar piepjes uit de luidspreker. Bij gebruik van het DOS blijven de vectoren #20C - #20F ongewijzigd. FAST en SLOW gebruiken de vectoren #214 - #217. Gebruik dus deze beide commando's met overleg.

Auto repeat

Na ongeveer 0.5 sec. worden alle toetsen met een snelheid van ongeveer 10 tekens per sec. herhaald. REPT versnelt dit nog aanzienlijk.

CoNTRLolled output

Het indrukken van CTRL+SPACE stopt de uitvoer naar het scherm bij het eerst volgende "nieuwe regel" teken (#0D) tot CTRL+SPACE opnieuw worden ingedrukt. Tijdens die pauze kunnen control commando's worden ingetoetst (bijv. CTRL+@ = inverteer het tekstwindow).

Editing

De gebruikelijke screen-edit mogelijkheden van de ATOM met COPY en de cursortoetsen kunnen worden uigevoerd. Uiteraard slechts indien de geselecteerde mode (32 of 42 t/regel, zwart of wit) gelijk is aan de mode waarin het file naar het scherm is geschreven. Niet herkende tekens (t.g.v. overlappende graphics of een niet passend gekozen mode worden in spaties omgezet. Terwijl tekens groter dan 6*8 pixels in gecomprimeerde mode (42 tekens) niet worden herkend in die mode.

Control codes

SCREENROM zendt op de gebruikelijke wijze control codes (PRINT$5 bijv.) naar de uitvoer op het scherm. Sommige control codes dienen onmiddelijk gevolgd te worden door een of meer parameters. (bijv. PRINT$31$20$12). Die control codes die geen extra parameter behoeven kunnen worden gebruikt in direct mode (CTRL+key). Er zijn geen ERROR meldingen na bijv. CTRL+L <RET>. Hieronder volgt een lijst van de mogelijke control codes met de daarbij behorende functies. Voorafgaand aan de code is de decimale waarde gegeven van die code (te gebruiken bij PRINT$x) en wordt gevolgd met de in direct mode in te drukken toets.

```
 0 CTRL-@ NUL  - wordt niet herkend.
 1             - het hierna volgende teken wordt naar de printer gezonden.
 2 CTRL-B STX  - enabled de printerpoort.
 3 CTRL-C ETX  - disabled de printerpoort.
 4 CTRL-D      - selecteert de 32 tekens per regel tekstmode.
 5 CTRL-E      - selecteert de 42 tekens per regel tekstmode.
 6 CTRL-F ACK  - start de output naar het scherm.
 7 CTRL-G BEL  - beep
 8 CTRL-H BS   - verplaatst de cursor 1 plaats naar links.
 9 CTRL-I TAB  - verplaatst de cursor 1 plaats naar rechts.
10 CTRL-J LF   - verplaatst de cursor 1 regel omlaag (indien geen pagemode is
                 geselecteerd scrolt de tekst)
11 CTRL-K VT   - verplaatst de cursor 1 regel naar boven (niet op de bovenste
                 regel)
12 CTRL-L FF   - maakt het tekstwindow schoon (zwart), zet de cursor links boven
                 in het window en (re)set de characterset op wit.
13 CTRL-M CR   - zet de cursor vooraan de regel
14 CTRL-N      - pagemode aan. Als de cursor de laatste regel van het
                 tekstwindow bereikt stopt de uitvoer totdat SHIFT is ingedrukt
                 waarna de uitvoer wordt hervat vanaf de top van het scherm.
15 CTRL-O      - pagemode uit. Het scrollen wordt hervat.
17 CTRL-Q      - inverteert het tekstwindow en de char.set.
18 CTRL-R      - inverteert het gehele scherm inclusief de char. set.
19 CTRL-S      - maakt het scherm schoon (zwart), zet de char. set op wit maar
                 verplaatst de cursor niet.
20 CTRL-T      - inverteert de character set.
21 CTRL-U NAK  - stopt de uitvoer naar het scherm totdat CTRL-F (ACK) wordt
                 ingetoetst.
22             - met aansluitend 4 bytes, bepaalt de graphics mode en de
                 tekstwindow afmetingen. Bij gebruik van de commando's MOVE,
                 DRAW, PLOT dient het juiste CLEAR commando gebruikt te worden.
                 (bijv. PRINT$22$2;CLEAR2 )
23             - gevolgd door 9 parameter bytes definieert uw eigen letterset.
                 Het eerste byte is de charactercode die u wenst te wijzigen. De
                 overige 8 bepalen dat nieuwe teken in het 8*8 pixel formaat te
                 beginnen van bovenaf. Let er op dat het meest significante bit
```

vrij moet blijven om de tekens van elkaar te kunnen scheiden en dat tevens het eerste byte 0 moet zijn i.v.m. de regelspatie. Bij de 42 t per regel mode kunnen slechts 6*8 pixels van elke byte worden getoond daarom moeten de 2 minst significante bits van elk byte 0 zijn.
(bijv. PRINT23#24$0$#44$#28$#7C$#10$#7C$#10)

24 CTRL-X - delete 1 regel, zet de cursor vooraan de regel
26 CTRL-Z - reset het tekstwindow naar het gehele scherm
27 ESC - escape
28 - definieert het tekstwindow met 4 aansluitende parameters voor links, onder, rechts, boven
voor 32 tekens; 0 tot 31 = links ---> rechts
voor 32 tekens; 0 tot 24 = boven ---> onder
bijv. PRINT$28$2$22$29$2 reduceert het tekstwindow met 2 tekens rondom.
30 CTRL-^ - plaatst de cursor in de linker bovenhoek van het tekstwindow.
31 - met 2 opvolgende bytes bepaalt de cursorpositie; respectievelijk horizontaal en verticaal gerekend vanaf de linker boven hoek. De eenheden zijn afgeleid van de gekozen tekstmode of afmeting. De navolgende tekst wordt na de nieuwe cursor positie geprint.
127 DEL - wist het voorgaande teken en plaatst de cursor 1 teken terug.

Ten slotte.
Met LINK#FF3F, de ATOM breakroutine, verlaat u de SCREENROM. LINK#AFF0 daarentegen verandert slechts de WRCVEC en de RDCVEC en gaat over naar de ATOM printmode. De tekstpagina pointer blijft dan op #2C00. De SCREENROM is niet compatibel met de colour graphics mode en het BBC kaartje. Voor het opheffen van de screennoise is uitsluitend een hardware oplossing noodzakelijk. Een software oplossing vertraagt de werking onaanvaardbaar.

PBIT

```
10 REM PBIT E.RONDA 27-11-84
20 G.120
30 DIT SOURCE-PROGRAMMA GENEREERT CODE OM EEN <EXPRESSIE>
40 ALS BITS TE PRINTEN.
50 SYNTAX: PBIT<EXPRESSIE1>,<EXPRESSIE2>,...
60 ALLEEN VAN EXPRESSIE1 WORDEN DE TWEE LAAGSTE BYTES ALS
70 BITS GEPRESENTEERD.
80
90 BV.: PBIT?#B001&#40'
100 00000000 01000000   40
110
120 J=7;DIMLLJ;F.I=0TOJ;LLI=#FFF;N.;@=0
130 P.$12"SOURCE PBIT"';DOP."=";U.C.>=12
140 IN.''"GEEF STARTADRES BV #3C00 "Z;IF?18=Z/256;G.140
150 P.$21;F.I=1TO2;P=Z;IFI=2P.$6''
160[
170:LL0 JSR#C78B;JSR#C3CB
180 LDA#53:JSRLL4;LDA#52;JSRLL4;JMP#C349
190:LL4 LDX@7:STA#66
200:LL5 BIT#66;BPLLL6;LDA@#31;BNELL7
210:LL6 LDA@#30
220:LL7 JSR#CA4C:ASL#66;DEX;BPLLL5;JMP#F379
230];N.;P.'"CODE VAN "&Z" TOT "&P'$3;E.
```

Hierna volgen enkele voorbeelden die een en ander wat zullen verduidelijken:

```
F.I= 1 TO 10;BPIT I';N.;P.''
00000000 00000001     1
00000000 00000010     2
00000000 00000011     3
00000000 00000100     4
00000000 00000101     5
00000000 00000110     6
00000000 00000111     7
00000000 00001000     8
00000000 00001001     9
00000000 00001010     A
```

```
F.I=1 TO 10;PBIT2*I';N.;P.''
00000000 00000010     2
00000000 00000100     4
00000000 00000110     6
00000000 00001000     8
00000000 00001010     A
00000000 00001100     C
00000000 00001110     E
00000000 00010000    10
00000000 00010010    12
00000000 00010100    14
```

```
DO;PBIT?#B001';U.0;P.''$3'
00000000 11111111     FF
00000000 11111111     FF
00000000 01111111     7F
00000000 01111111     7F
00000000 11111111     FF
00000000 11111111     FF
00000000 10111111     BF
00000000 10111111     BF
00000000 10111111     BF
00000000 11111111     FF
```

INDEX MAKER VOOR TAPES

Met het onderstaande programma is een index te maken voor al Uw computer tapes.
Dit programma kan de index ook naderhand weer wijzigen.
Voorbeeld:

```
---------------------------------
:          INDEX TAPE 1          :
:                                :
:  SOURCEMAKER    2900 3B2E C2B2 :
:  ATOMCALC       A000 B000 A000 :
:  SCHIPHOL       2900 3F00 C2B2 :
:  SPACE WAR      2900 358A C2B2 :
:                                :
:                                :
:                                :
:                                :
:  LOAD"SOURCEMAKER"             :
:  *LOAD"ATOMCALC"               :
:  *RUN"SCHIPHOL"                :
:  *RUN"SPACE WAR"               :
:                                :
:                                :
---------------------------------
```

De index is van tape te lezen met behulp van *LOAD"INDEX" .

```
 10 REM CASSETTE INDEX MAKER
 20 REM F.SPENKELINK
 30 REM NIJVERDAL
 40
 50 REM INDEX LADEN MET
 60 REM *LOAD"INDEX"
 70
 80 REM dim. 15 strings
 90
100 DIMA32,B32,C32,D32,E32,F32,G32,H32,I32
110 DIMJ32,K32,L32,M32,N32,O32
120 !#80=#85FFE320;!#84=#608A
130
140 REM print menu
150
160 GOS.z
170 P.'" INVOEREN INDEX        1"''" AFDRUKKEN INDEX        2"
180 P.''" SAVEN 1200  BAUD      3"''" SAVEN  300  BAUD       4"
190 P.''" EDITEN   INDEX        5"''" LADEN    INDEX         6"
200 GOS.y;LI.#80
210
220 REM welke keuze
230
240 IF ?#8A=49 GOS.a
250 IF ?#8A=50 GOS.b
260 IF ?#8A=51 OR ?#8A=52 GOS.c
270 IF ?#8A=53 GOS.d
280 IF ?#8A=54 GOS.e
290 G.160
300
```

```
310 REM invoeren index
320
330aGOS.z
340 P.''" U KUNT NU 15 REGELS TEKST"'" INVULLEN"
350 P.''" TOTAAL 32 KARAKTERS PER REGEL"
360 P.''" LIEFST TWEE SPATIES VOORAAN"''" DRUK OP EEN TOETS"
370 GOS.y;LINK#FFE3
380 P.$12;IN.$A$B$C$D$E$F$G$H$I$J$K$L$M$N$O
390 ?#E1=0;R.
400
410 REM afdrukken index
420
430bP.$12;?#E1=0
440 P.$A'$B'$C'$D'$E'$F'$G'$H'$I'$J'$K'$L'$M'$N'$O
450 LI.#FFE3;R.
460
470 REM saven index
480
490cGOS.z
500 P.'" U SAVED NU DE GEMAAKTE INDEX"
510 P.''" ZET DE RECORDER OP RECORD"
520 P.''" DRUK TWEEMAAL OP DE SPATIEBALK"
530 P.''" DE INDEX KOMT OP HET SCHERM"''
540 P." NA DE PIEPJES OP DE SPATIEBALK"
550 P.'" DRUKKEN OM HET SAVEN TE STARTEN"
560 GOS.y;LINK#FFE3
570
580 REM 300 of 1200 baud
590
600 IF ?#8A=51 FC.
610 IF ?#8A=52 SC.
620 GOS.b
630 *NOMON
640 P.$7$7$7
650 *SAVE"INDEX"8000 8200
660 *MON
670 R.
680
690 REM edit routine
700
710dP.$12;GOS.b;P.$30;?#E1=#80
730 IN.$A$B$C$D$E$F$G$H$I$J$K$L$M$N$O
740 R.
750
760 REM inlaad routine
770
780eGOS.z
790 P.''" U KUNT NU EEN INDEX VAN BAND"
800 P.'" INLADEN, NA HET INLADEN STAAT"
810 P.'" DE ATOM IN INPUTMODE ZODAT"
820 P.'" MET COPY DE INDEX WEER IN HET"
830 P.'" GEHEUGEN WORDT GEPLAATST"
840 P.'" VOOR VERDERE BEWERKING"''''''
850 FC.;?#E1=#80
860 *LOAD"INDEX"
870 P.$30;IN.$A$B$C$D$E$F$G$H$I$J$K$L$M$N$O
880 R.
890zP.$12;?#E1=0;P." *** CASSETTE INDEX PROGAMMA ***";R.
900yP.''" ********************************";R.
```

WORDPACK GERELOCATED

Aangezien ik een P-CHARME eprom heb op IC-24 en geen (mini)schakelkaart, kon ik WORDPACK niet gebruiken. Dit vond ik jammer, dus heb ik WORDPACK gerelocated.
Op zijn zachst gezegd was dit rot werk want ik had het RELOC statement van de JOSBOX niet tot mijn beschikking, dus moest ik het met de hand doen. Achteraf gezien was dit eigenlijk best wel gunstig omdat ik dan meteen wist waar de data tabellen stonden. Ik heb dus WORDPACK naar #7000 gecopieerd, en ben toen begonnen met het aanpassen van de adressen.
Als je WORDPACK relocate dan moet hij op een #X000 pagina beginnen. Dus niet op bv. #3C00 maar op bv. #4000.Ik leg zo wel uit waarom.
Ik beschrijf deze procedure nu zodat de WORDPACK naar #7000 gaat.
Vanaf #A002 t/m #A07A staat gewoon machine code die je gewoon kan relocaten, M.U.V. :

```
#A067 LDA @#29
#A069 STA #0202
#A06C LDA @#A5
#A06E STA #0203
```

Zoals je ziet wordt hier de BRK-vector gewijzigd, en wijst nu naar #A529. Dit moet dus #7529 worden, dus maak van die LDA @#A5 maar LDA@#75.
Vanaf #A07B t/m #A084 staat een klein tabelletje, deze kan je gewoon naar #707B toe copieren.
Vanaf #A085 t/m #A62C staat ook weer gewone machine code, die kan je dus ook weer zonder probleem relocaten naar #7085.
Vanaf #A62D t/m #A686 staat weer een tabel. Copieer deze eerst naar #762D. Vanaf #764B t/m #7668 staan er 31 bytes die met een 'A' beginnen (dus #AX). Deze A's moeten in 7's veranderd worden zodat de adressen in deze tabel weer kloppen.
Vanaf #7669 t/m #7686 staan nog 31 bytes die NIET veranderd moeten worden.
Vanaf #A687 t/m #AABA staat weer gewoon machine code, relocaten maar.
Vanaf #AABB t/m #AAC4 is er weer een tabel. Deze moet je ook gewoon weer copieren naar #7ABB.
Vanaf #AAC5 t/m #ACEA weer gewone machine code.
Nu weer een tabelletje:

```
#ACEB CE AC 7F AC
#ACEF 52 FE E1 AA
#ACF3 15 AB 94 FE
```

De AC,AC,AA en AB moet je dus veranderen in 7C,7C,7A en 7B.
De volgende 9 bytes zijn 3 JMP's, dus wel relocaten.
Nu, vanaf #AD00 t/m #AFFF volgt de karakter tabel, deze moet je dus ook normaal KOPIEEREN naar #7D00.
De WORDPACK print routine gebruikt een nogal vernuftig truukje om de karakters te decoderen naar hun plaats in de karakter data tabel. Ik heb hier over dus een paar avonden over zitten piekeren voordat ik het door had.

Op #ABA5 staat:

```
#ABA5 LDA @#2B
```

In binair is #2B : %00101011. Als je dit nou 2 plaatsen naar links schuift dan staat er #AD , of m.a.w. de pagina waar de karakter tabel begint. Als wij van die 'A' een 7 maken dan begint de tabel vanaf #7D00. De tabel MOET op een #XD00 pagina beginnen. (Als je de tabel ergens anders wil laten beginnen dan moet je

de WORDPACK printroutine veranderen, maar daar had ik geen zin in).
Die %00101011 wordt dus %00011111, of te wel die LDA @#2B wordt nu LDA @#1F.

Nou nog het laatste :

```
#A57F LDA @#71
#A581 STA #1C
#A583 LDA @#A0
#A585 STA #1D
```

Hier wordt het executie adres van de te saven WORDPACK file gezet op #A071. Als je wilt kan je hier #7071 vanmaken maar het hoeft niet.
Het start adres voor EDIT is nu #7030.
Het start adres voor TEXT is nu #7B05.
Je kan deze statements ook voor P-CHARME herkenbaar maken.Hier volgt een hexdump van hoe dat er uit ziet in het geheugen:

```
7000 FF E3 C6 45  . . . E
7004 44 49 54 F0  D I T .
7008 30 54 45 58  0 T E X
700C 54 FB 05 B0  T . . .
```

Je kan de machine code die er eerst stond (tot aan #702F) overschrijven want dit is de interpreter van WORDPACK en die wordt dan dus niet meer gebruikt.

Als je de volgende veranderingen (van Arie Marchal A.N. 3.5) ook nog aanbrengt, dan gaat het ook goed als je op Q drukt (als je dus Quit van de tekstverwerker):

geheugen plaats	was	wordt
#7CF1	#E1	#94
#7CF2	#7A	#FE
#7AD9	#15	#07

Nou volgens mij moet WORDPACK het nu doen op #7000. Bij mij doet hij het tenminste wel.

Een WAARSCHUWING!!! als je een lange WORDPACK file hebt dan kan deze ove de #7000 gaan en dus WORDPACK op #7000 overschrijven.Zorg er dus voor dat een WORDPACK file niet langer is dan 28 WORDPACK pagina's.
In Acorn Nieuws 3.6 staat nog een interessant artikel, van Henk Boxma, over printer codes vanuit WORDPACK.

Wist U trouwens al dat er een nieuwe tekstverwerker rom in Engeland verkrijgbaar is voor de ATOM. Het is de 'BEARSOFT EDITOR' van BEARSOFT. Deze rom is een verbetering van de oude WORDPACK van ACORNSOFT.Hij kost alleen 35 pond incl. btw (FL 140,- of zoiets), toch wel een beetje duur niet waar.
Nou veel plezier met de WORDPACK op #7000.

BBC KAARTJE

De laatste tijd is het ACORN BBC kaartje voor het bedrag van maar F 79,- in de winkel te vinden. Dat zal wel voor het laatst zijn want de laatste ATOM-waren moeten uit de verkoop. Ik kon het voor dit bedrag niet laten, om ook zo'n kaartje te kopen. Na alles wat hierover eens geschreven was, gelezen te hebben, ben ik eens gaan bekijken hoe ik het beste dit kaartje kon inbouwen. Het kaartje neemt in de ATOM kast nogal veel plaats in zodat andere kaarten als 16K kaart en schakelkaart moeten verdwijnen. Behalve dat het kaartje fysiek veel ruimte vraagt, haalt het op een dermate walgelijke wijze de ATOM overhoop dat het actief laten meedraaien van andere kaarten op de ATOM moeilijkheden geeft. Ik heb daarom (voor de toekomst, want ik verbouw nogal veel aan mijn Atom) het kaartje maar eens goed bestudeerd.
In eerste instantie moeten in de Atom alle kaarten, gestapelde geheugens en schakelkaarten verwijderd worden, om met behulp van dit kaartje BBC-basic te kunnen draaien. Dit is natuurlijk niet acceptabel, en daar moeten wat verandering in worden aangebracht. Voor de TELEC schakelkaart, de 16K kaart op #4000 en alle gestapelde varkens heb ik een oplossing. Met behulp van een nadere beschrijving moet het mogelijk zijn zelf ook aanpassingen te kunnen maken.
Zelf heb ik een TELEC-schakelkaart, een lang en smal geval, die een plaatsje kan vinden tussen het Atom board en de bovenkant van de kast. Met behulp van een lang stuk flatcabel is deze weer verbonden met de ic 24 connector van de Atom board. Echt prettig is dit niet, want de Atom bus is op deze plaats niet gebufferd, en de lange flatcabel kan wel eens teveel zijn (te zware capacitaire belasting). Bij mij functioneert het echter goed. Deze methode is ook geschikt voor de minischakelkaart. Andere grotere kaarten moeten helaas buiten de Atom geplaatst worden.
De 16K kaart op adres #4000 - #8000 zit bij mij buiten de kast, in een mooi hiervoor geschikt plat doosje. Verder heb ik gestapelde geheugens vanaf adres #3C00 - #4000 en #9800 - #A000. Deze zitten de BBC kaart fysiek ook wat in de weg, want die komt boven deze ic's te liggen. Bovendien accepteert de BBC kaart geen geheugen tussen #2000 - #2800 en #3C00 - #40C0. Toch willen we in 'Atom mode' straks wel geheugen hebben van #2000 - #4000. Dat kan ! Als eerste moeten de gestapelde varkens van #2000 - #2800 met hun selectielijnen aan de +5 volt gelegd worden. Deze werken dan niet meer. De BBC kaart kan in de voeten van ic's 24, 23, 6 en 8 gestoken worden. Omdat de kaart door de varkens iets hoger moet komen te liggen, kun je het beste op de 4 connectors van het kaartje 4 passende ic-voetjes plaatsen. Dan past de kaart nog net in de kast en kunnen de varkens tussen BBC kaart en Atom board blijven zitten! Bovendien zijn de pootjes van de kaart iets te dik, zodat ze de voetjes op de Atom kaart kapot kunnen drukken. Op deze manier opgelost geeft het later geen teleurstellingen.
Blijft nog het geheugen van #3C00 - #8000 over. De varkens voor #3C00 moeten met hun selektielijn aan pen 7 ic 9 van het BBC boardje gelegd worden. Het is verstandig deze ook door te trekken naar pen 1 ic 5 van het Atom boardje, om de bus dan dicht te trekken. Hiervoor moet pen 1 ic 5 natuurlijk wel van de print losgenomen worden. Ic 9 van het BBC boardje vervangt het ic dat vroeger op de Atom board voor de selektie van #3CXX gebruikt werd. Nu moet pen 6 van ic 12 van het BBC kaartje van de print losgemaakt worden en aan de +5 volt gelegd worden. We hebben nu geheugen van #2000 - #4000.
Blijft nog de 16K kaart over. Hiervoor kan het truukje van Gerard Morsink gebruikt worden: pen 11 ic 14 BBC board moet van de print losgemaakt worden, en met pen 12 van ic 12 BBC board verbonden worden. De externe bus gaat nu in de 'Atom mode' open op de adressen #4000 - #8000.
Alleen mensen met een gewone schakelkaart, disk controller, of een 16K kaart op #0000 - #4000 moeten nu zelf nog aanpassingen verzinnen.
Daarvoor zal ik verderop beschrijven wat het kaartje nu precies doet, en wat het effect van de net uitgevoerde handelingen is.
Verder heb ik nog wat subtiele wijzigingen aangebracht. Het schakelen tussen een 'BBC machine' en een Atom vanaf het toestenbord gebeurt met control-break en

shift-break. Control-break plaatst de nu ontstane 'dubbel-computer' altijd in de Atom mode, en shift break-altijd in de BBC mode. Nu waren sommige al bekend met het indrukken van shift-break in plaats van gewoon break, om per ongeluk indrukken van de break-toets te voorkomen. Het zou dus logisch zijn om shift-break te laten corresponderen met Atom mode, en control-break met BBC mode. Dit kunnen we doen door de control en shift lijnen vanaf het BBC boardje naar de atom met elkaar te verwisselen.
Verder is het zo dat bij mij in de BBC stand na power-up opgestart wordt. De BBC mode is voor mij echter optioneel, en ik heb dus liever dat na power-up de atom mode de voorkeur heeft. Dat kan afgedwongen worden door pen 5 ic 15 (BBC board) met een heel klein condensatortje met de +5 volt te verbinden. De flipflop die met ic 16 en 15 gevormd wordt, krijgt zo een voorkeur voor de Atom mode.
Rest nog een beschrijving van wat het kaartje nu eigenlijk doet zodat zelf ook wijzigingen aagebracht kunnen worden.
Het plaatsen van het BBC kaartje verandert al direct wat aan de geheugenindeling. De vogende verschuivingen vinden plaats:

Voorheen:	Na inbouw in Atom mode:	
#3800	#2000	
#3400	#2400	
#3000	#2800	
#2C00	#2C00	
#2800	#3000	
niet aanwezig	#3400	2 extra 2114's
niet aanwezig	#3800	2 extra 2114's
#3C00	#3C00	onze oude varkens !

Er vindt dus een aanzienlijke verschuiving plaats. De 4 2114 van het BBC kaartje nemen nu plaats op #3400 - #3C00. Daarom moeten we dus ook oude varkens van #2000 - #2800 onschadelijk maken. Ze worden door het BBC kaartje automatisch vervangen. De varkentjes op #3C00 blijven op hun plaats. De aangebrachte veranderingen zorgen ervoor dat in BBC mode de zero-page ic's 51 en 52 niet meer werkzaam zijn, maar vervangen worden door de ic's die in Atom mode #2000 bedienen. Hierdoor hebben we in Atom mode nu geheugen van #2000 - #4000.
Verder is van belang dat het BBC kaartje geen adreslijnen verandrt, zodat bijvoorbeeld geheugen tussen #400 en #2000 dubbel geadresseerd kan worden, met alle gevolgen van dien. Iets dergelijks geld met name voor de 16K kaart, die zelf selekteert van #4000 tot #8000. Ik kom hier nog op terug.
In de Atom mode is de externe bus nu gesloten in de gebieden:

#0000 - #0400	
#2000 - #3C00	#3C00 kan nog afgeknepen worden door selektie lijn door te trekken naar pen 1 ic 5 Atom board
#8000 - #E000	
#F000 - #FFFF	

Door de wijziging van Gerard Morsink is de bus bij de adressen #4000 - #8000 wel geopend. Bij het omschakelen naar BBC mode verandert er nu het een en ander:

ATOM MODE:	BBC MODE:	
#0000 - #0400	Uitgeschakelt	
#0400 - #2000	#0400 - #2000	dubbel voorkomen
#2000 - #4000	#0000 - #2000	workspace en tekst ram
	#2000 - #4000	nog te gebruiken
#4000 - #8000	#4000 - #8000	afgesloten in BBC mode zelfs op de externe bus
#8000 - #C000	#4000 - #8000	Display, pia en via en A-blok
Niet aanwezig	#8000 - #C000	BBC rom
#C000 - #D000	#C000 - #D000	wordt vervangen door MOS extension rom
#D000 - #E000	#D000 - #E000	verdwaalde fl point rom
#E000 - #F000	#E000 - #F000	Disk ??????
#F000 - #0000	#F000 - #0000	wordt vervangen door BBC MOS rom

In BBC mode is de externe bus nu gesloten voor:
#0000 - #2000
#4000 - #E000
#F000 - #0000
Doordat de BBC graphics ruimte nu op #4000 - #6000 zit, wordt ook de 16K kaart aangesproken. Dat komt omdat deze kaart zelf seleteert, en de adres lijnen niet veranderd worden in BBC mode. De externe bus is echter dichtgeknepen, zodat de 16K kaart geen informatie kan doorspelen. Hij is echter wel geselekteerd, en verandrt dus van inhoud. Een echt fraaie oplossing is dit dus niet.
We zouden het een en ander naar onze hand kunnen zetten. Enige ideeen, maar nog niet uitgewerkt zijn:

- In BBC mode de r/w lijn bij #4000 - #8000 altijd op read zetten. De 16k kaart zal dan nooit lezen, en dus van inhoud veranderen.
- Wat adreslijnen omgooien in BBC mode, zodat ook #2000 - #4000 als tekst ruimte tot onze beschikking komt. #2000 - #4000 moet dan #4000 - #6000 worden. De 16K kaart kent deze adressen al.

Helaas moet voorlopig in BBC mode de 16K kaart maar op write-protect gezet worden. In het korte bestaan van mijn BBC basic, heb ik alvast wat beslag weten te leggen op de volgende informatie:

Vectoren:	doel:	waarde:
#0200	gereserveerd	#FFFF
#0202	BRK	#B42D
#0204	IRQ	#8000
#0206	gereserveerd	?????
#0208	command line interpreter	#F775
#020A	BYTE	#F0D2
#020C	WORD	#F000
#020E	write char	#F54F
#0210	read	#FE6D
#0212	load/save	#F71A
#0214	gereserveerd	?????
#0216	BGET	#FBC1
#0218	BPUT	#FC51
#021A	gereserveerd	?????
#021C	FIND	#FC0B

Van deze vectoren kan gebruik gemaakt worden door onder anderen een eprom te plaatsen op #C000 met als eerste byte #AA. Na een brk via het toetsenbord volgt een JSR #C001, zodat we elegant het systeem kunnen wijzigen.
Wist U verder dat:

- Er fouten in het bijbehorende boekje staat.
- Er door een engelse firma waarschijnlijk toch nog een MOS extension rom wordt gemaakt.
- Er veel elegantere manieren zijn om BBC basic te verkrijgen.
- Dat hier waarschijnlijk nog eens over geschreven zal worden.
- Er behoorlijke fouten zitten in de BBC rom, en MOS rom.
- Deze er binnenkort uitgehaald zullen worden.
- Er misschien binnen onze club ook een MOS extension rom gefabriceerd zal worden.
- Het laatste woord over BBC basic nog niet gesproken is.
- Ik er graag alles over zou willen vertellen.
- Deze 'tweede machine' ook voor mij compleet nieuw is.
- We dus erg blij mogen zijn met onze goed gedocumenteerde Atom.

Veel succes !

TEST UTILITY SOURCE

Dit statement is speciaal geschreven voor opgenomen te worden in een schakelsoft of in een jumper op de 16K ramkaart. Het statement kan bijvoorbeeld aangeroepen worden met HELP en is vergelijkbaar met gelijknamige statement uit de romarbox. Men moet er echter wel voor zorgen dat bij gebruik in een schakelsoft eerst de interpreter van de schakelsoft wordt doorlopen voordat deze in de utility's gaat zoeken en bij de romar-box terecht komt. Geeft dit problemen dan zal men een nieuwe naam moeten bedenken bijvoorbeeld "TEST" of iets dergelijks.

Dit nieuwe statement biedt de mogelijkheid om het herkennen van de voorgeschakelde utility-rom te vereenvoudigen. Bij het aanroepen van deze "help" worden alle namen inclusief executie-adres op het beeldscherm zichtbaar gemaakt.

Om dit statement te kunnen gebruiken moet de interpreter van de utility direkt beginnen op #A002 en de statement tabel beginnen op #A032. Voorbeelden van deze zijn: Romar-box, AXR-1, ASBK-1 enz. Is er geen utility aanwezig dan zal er de melding "NO UTILITY" op het scherm verschijnen.
Verder nog veel plezier met dit statement.

```
 10REM TEST UTILITY
 20REM***********************
 30REM*    CHARL DE MOOR    *
 40REM*      OKT  1984      *
 50REM*     045  272358     *
 60REM***********************
 65DIMNN(30),A5,B5;F.I=0 TO 6;NN(I)=#FFF;N.
 70P.$12"TEST UTILITY "
 80IN.''"ASSEMBLING TO: "Z;IF?18=Z/256;G.80
 90IN."PRINTER (Y/N): "$A
100IN."LISTING (Y/N): "$B
110P.'"pass 1"'$21;F.I=1TO2;P=Z;IFI=2;P.$6$11"pass 2"'''$21
120?#E8=0;IFI=2;IF?A=89P.$2
130IFI=2;IF?B=89P.$6
140[
150:NN1;LDA#A001;CMP@#BF;BEQNN2
160JSR#FFED;JSR#F7D1;];$P="NO UTILITY";P=P+LENP;[NOP;JMP#C2CF
170:NN2;JSR#F7D1;];!P=#0C06;P=P+2;$P="CHECK UTILITY";P=P+LENP
180!P=#0D0A;P=P+2;[;LDX@#00;STX#E1;JSR#F7FD
190:NN3;LDY@0;JSR#FFED
200:NN4;LDA#A032,X;BMINN5;JSR#FFF4;LDA#B001;AND@#20;BEQNN6
210INX;BNENN4
220:NN5;CMP@#C5;BEQNN6;LDA@#0A;CPY@0;BEQP+4;LDA@#18
230STA#E0;INX;INX;LDA#A030,X;JSR#F802;LDA#A031,X;JSR#F802
240CPY@#01;BEQNN3;JSR#F7FD;JSR#F7FD;JSR#F7FD;INY;JMPNN4
250:NN6;JSR#FFED;JSR#FFED;LDA@#80;STA#E1;JMP#C2CF
260];N.;IF?#E8=#23;G.280
270P.$6''"READY"''" START: "&Z'" END  : "&P'';@=8;E.
280P.$6$7$7$7"OUT OF RANGE DETECTED"'';@=8;E.
```

EEN CASE-STRUCTUUR IN FORTH.

INLEIDING:

Het in de P-charme voorkomende case-statement levert overzichtelijke, meervoudige vertakkingen in een programma op. Het is bijvoorbeeld erg handig voor het verwerken van een keuze uit een menu.

En dergelijke structuur is ook mogelijk in FORTH.
Hierbij worden de woorden CASE, OF, ENDOF en ENDCASE gebruikt.
Definities hiervan vond ik in het tijdschrift Dr. Dobb's Journal. Ze zijn oorspronkelijk geprogrammeerd door Charles E. Eaker en gepubliceerd in FORTH Dimensions vol.II no.3 p.37.
De definities zijn door mij iets aangepast aan Atom FORTH.

GEBRUIK:

De hele case-structuur moet in een colon-definitie worden opgenomen (die ik hieronder NAAM genoemd heb), want anders volgt er een foutmelding.

De opbouw van de case-structuur is als volgt:

```
: NAAM
CASE
waarde  OF  aktie  ENDOF
waarde  OF  aktie  ENDOF
   -     -    -      -
   -     -    -      -
waarde  OF  aktie  ENDOF
aktie-alternatief
ENDCASE
      ;
```

TOELICHTING:

Bepalend voor de keuze van de vertakking is de bovenste waarde op de stack op het moment waarop CASE uitgevoerd wordt. Deze waarde wordt vergeleken met de waarde die in iedere vertakking voor OF staat. Zodra een gelijke waarde wordt gevonden, wordt deze vertakking uitgevoerd en daarna naar ENDCASE gesprongen.
Als geen van de waarden in de vertakkingen overeenkomt met de testwaarde, dan wordt (indien aanwezig) de aktie tussen de laatste ENDOF en ENDCASE uitgevoerd.

NAAM staat voor de naam van de colondefinitie waarin de case-structuur voorkomt.
Wat ik "aktie" heb genoemd kan bestaan uit een of meer FORTH woorden. Het overzichtelijkst is het waarschijnlijk alle aktie binnen een vertakking als een woord te definieren.
Zorg er in ieder geval wel voor dat de totale aktie binnen een vertakking of "aktie-alternatief" stack-neutraal zijn, d.w.z. geen extra waarden van de stack halen of er op achterlaten, anders treden er fouten op. Waarden die vanuit een "aktie" doorgegeven moeten worden kunnen dus het beste in een variabele gestopt worden. Dit is overzichtelijker ook.

Alle FORTH woorden die tussen NAAM en CASE staan worden altijd uitgevoerd, evenals alle woorden die tussen ENDCASE en ; (semicolon) staan.

Let er wel op dat de te testen waarde boven op de stack moet staan op het moment dat CASE uitgevoerd wordt. Deze waarde kan naturlijk al op de stack staan voordat NAAM uitgevoerd wordt, maar dan mag het eventuele gedeelte tussen NAAM en CASE de te testen waarde niet aantasten. Een andere mogelijkheid is dat de te testen waarde tijdens de uitvoering van het gedeelte tussen NAAM en CASE uit een variabele gehaald wordt en op de stack gezet.

De waarden die voor OF in de vertakkingen staan kunnen als getallen in de definitie van NAAM opgenomen worden. De namen van, van te voren gedefinieerde, constanten kunnen als "labels" voor OF in de vertakkingen gebruikt worden, waardoor de leesbaarheid van de structuur groter wordt. Bij het compileren van NAAM worden dan de waarden van de constanten in de definitie opgenomen. Overal waar sprake is van getallen of waarden worden enkele precisie integer getallen bedoeld.

DE DEFINITIES:

```
SCR # 300
0 ( CASE CH.E.Eaker  FORTH Dimenions II, 3 p.37                    )
1 ( Bewerkt voor Atom FORTH door Sieb Spannenburg                    )
2 FORTH DEFINITIONS DECIMAL
3 : CASE       ?COMP CSP @ !CSP 4 ; IMMEDIATE
4 : OF         4 ?PAIRS COMPILE OVER COMPILE = COMPILE
5              [ HEX 28A2 , DECIMAL ]  HERE 0 , COMPILE
6              DROP 5 ; IMMEDIATE
7    -->

SCR # 301
0 ( CASE 2 S.SP. 22.11.84 )
1
2 : ENDOF      5 ?PAIRS COMPILE [ HEX 288D , DECIMAL ]
3              HERE 0 , SWAP 2  [COMPILE] THEN 4 ; IMMEDIATE
4
5 : ENDCASE    4 ?PAIRS COMPILE DROP BEGIN SP@ CSP @ = 0=
6              WHILE 2 [COMPILE] THEN REPEAT CSP ! ; IMMEDIATE
7
```

EEN VOORBEELD:

Als voorbeeld heb ik het hoofdmenu van DBASE 3.00 uit Acorn nieuws 1.1985 genomen. Het voorbeeld zet het menu op het scherm en wacht daarna de keuze af. Als een goede toets ingedrukt word, dan wordt de naam van de gekozen aktie afgedrukt, om te laten zien welke vertakking gekozen is. (Bij een volledige toepassing zou natuurlijk het gekozen programmagedeelte worden uitgevoerd.) Bij een niet geldige keus gebeurt er niets.

```
SCR # 501
0 ( VOORBEELD CASE  S.SP. 15.2.1985 )
1 : CREATF     CR ." CREATE     (DUMMY) " ;
2 : LOADF      CR ." LOAD       (DUMMY) " ;
3 : COS/DOS    CR ." COS/DOS    (DUMMY) " ;
4 : DIRECTORY CR ." DIRECTORY (DUMMY) " ;
5 : QUITF      CR ." QUIT       (DUMMY) " ;
6 : CLS  12 EMIT ;  0 VARIABLE  VLAG
7 -->
```

```
  SCR # 502
0 ( VOORBEELD CASE 2 )
1
2 : DOE-MENU   1 VLAG !    CASE
3   67 OF CREATF       ENDOF        60 OF LOADF        ENDOF
    42 OF COS/DOS      ENDOF        46 OF DIRECTORY    ENDOF
5   81 OF QUITF        ENDOF
    0 VLAG !
6  ENDCASE  ;
7 -->

  SCR # 503
0 ( VOORBEELD CASE 3 )
1 : .MENU  CLS  10 SPACES  ." MAIN-MENU " CR CR
2 ." 1...CREATE NEW DATABASE          (C) " CR
3 ." 2...LOAD EXISTING DATABASE       (<) " CR
4 ." 3...EXECUTE COS/DOS COMMAND      (*) " CR
5 ." 4...DATAFILE DIRECTORY           (.) " CR
6 ." 5...QUIT                         (Q) " CR CR
7 ." :PLEASE ENTER COMMAND  "  ;      -->

  SCR # 504
0 ( VOORBEELD CASE 4 )
1
2 : MENU    .MENU
3    BEGIN
4      KEY   DOE-MENU  VLAG @
5    UNTIL
6 ;
7
```

EEN TOEGIFT:

Er zijn misschien mensen die de IF ... ELSE ... THEN constructie in FORTH verwarrend vinden, en IF ... ELSE ... ENDIF iets makkelijker.
Zij kunnen ENDIF als volgt definieren:

```
  : ENDIF  [COMPILE] THEN ; IMEDIATE
```

Veel succes met Forth!

TEXT STATEMENT VOOR BBC BASIC

Naar aanleiding van het spoolerprogramma uit ACORNNIEUWS 7 jaargang 3 voor BBC basic, heb ikzelf een betere spooler geschreven. Het verschil tussen deze 2 programma's wordt al snel duidelijk bij gebruik. Bij dit programma, wat hetzelfde doet als het text statement uit wordtpack, wordt na aanroepen van dit programma alle output tevens in het textgeheugen gezet vanaf #2902. Tussen #2800 en #2902 worden spaties geplaatst zodat de BRK die we straks verplicht zijn uit te voeren om weer in ATOM mode te komen, niet de tekst modificeert. Werkwijze: roep dit programma aan, en list bijvoorbeeld een BBC programma. Ga daarna met BRK naar de ATOM mode en bekijk de text met behulp van WORDPACK. Alle tokens zijn nu vervangen door gewone ascii characters, dus kunnen we de file bijvoorbeeld vanuit de ATOM mode op de printer zetten.

```
 10 REM TEXT STATEMENT
 20 REM TEXT TO &800
 30 REM LIKE WORDPACK TEXT
 40 OSBPUT=&FFD4
 50 OSWRCH=&F54F
 60 WRCVEC=&020E
 70 LF=10
 80 REGISTER=&80:TEMP=&82
 90 TEXTSPACE=&902
100 FOR I%=&800 TO TEXTSPACE
110   ?I%=&20
120 NEXT
130 !REGISTER=TEXTSPACE
140 ?TEXTSPACE=4
150 DIM CODE%30
160 FOR PASS%=0 TO 3 STEP 3
170   P%=CODE%
180   [
190   OPT PASS%
200  .TEXT
210   CMP#LF
220   BEQ SKIP
230   STY TEMP
240   LDY #0
250   STA (REGISTER),Y
260   INC REGISTER
270   BNE P%+4
280   INC REGISTER+1
290   PHA
300   LDA #4
310   STA (REGISTER),Y
320   PLA
330   LDY TEMP
340  .SKIP
350   JMP OSWRCH
360   ]
370 NEXT PASS%:CLS:PRINT"TEXT TO &2800 ATOM MODE"
380 WRCVEC?0=TEXT AND &FF:WRCVEC?1=TEXT DIV 256
390 END
```

PLANETARIUM

Planetarium is een computersimulatie van ons zonnestelsel.
Het programma geeft nauwkeurig uitsluitsel over plaats en baan van alle bekende planeten in ons zonnestelsel, over een tijdspanne van zo'n honderd eeuwen.
Speciale aandacht is besteed aan het aarde-maan syteem,meteoorsporen en komeetbanen.

Na de opstart van het programma wordt een testbeeld gegenereerd waarmee de werking van T.V. of monitor gecontroleerd kan worden. Daarna volgt het menu.

Menu.
Het menu kent 8 onderdelen.

a "?" algemene informatie.
de "?" is een waardevolle toets in dit programma. Elke routine heeft namelijk eeen eigen infoblok dat men tijdens het draaien van de routine m.b.v. "?" kan oproepen. De routine gaat daarna verder waar zij was gebleven.

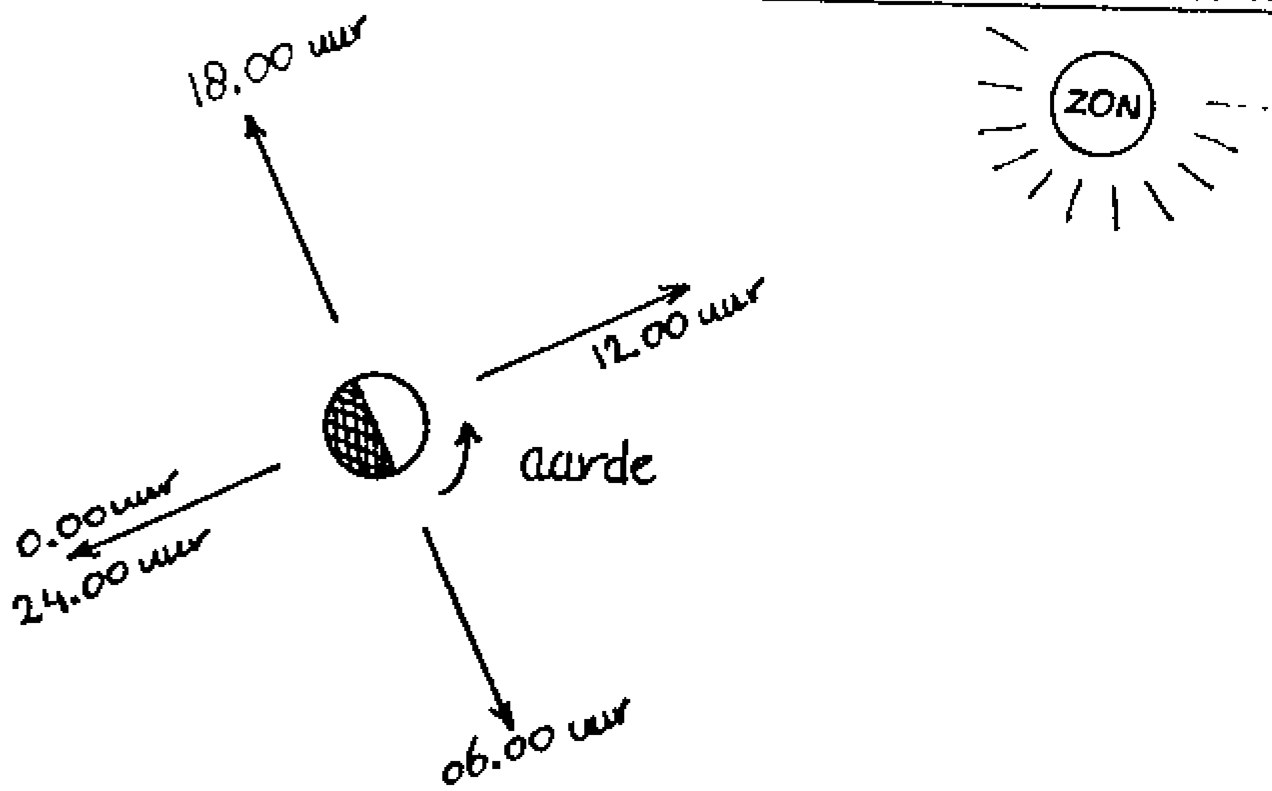
b "~" einde
Het programma neemt afscheid van de gebruiker, blijft in de hoofd textspace en schakelt de page-mode in.

c "1" de binnenste planeten
Mercurius,Venus,Aarde en Mars

d "2" de buitenste planeten
Jupiter,Saturnus,Uranus,Neptunus en Pluto
Deze splitsing is aangebracht vanwege de gigantische afmeting van ons zonnestelsel.
Routine "1" heeft als schaal 2,8 miljoen km./pixel, routine "2" 50,5 miljoen km./pixel!

e "3" aarde en maan
De baan van de maan om de aarde is in deze routine voor de duidelijkheid 400* vergroot.
De routines "1","2" en "3" geven een "plattegrond" van de planeetbanen, waarbij men steeds op het noordelijk halfrond van de aarde kijkt. Het referentiepunt is het zogenaamde lentepunt waarvan de richting op het scherm is aangegeven.

f "4" planetenzichtbaarheid
Deze routine laat zien waar de maan en de planeten van ons zonnestelsel te zien zijn vanaf een bepaalde plaats (of uur) op aarde.
U krijgt een rechthoekig scherm te zien met witte zijranden (de zon). Horizontaal staat de tijd op aarde, geldend op de datum die rechts onder in beeld staat.
Op het scherm worden de maan en de planeten geprojecteerd, elk vergezeld van een stipje vlak boven de onderlijn. Deze stipjes geven aan waar op een bepaalde dag, op een bepaald uur de hemellichamen te zien zijn, mits men "recht" het heelal inkijkt (langs de azimuth).
"planetarium" rekent met het plattegrondmodel, en kan dus wel tonen in welke richting een planeet staat gezien vanaf de aarde, maar niet op welke "hoogte".
Ter verduidelijking van de relatie tussen de tijd op aarde en de richting waarin U dan het heelal inkijkt:



"planetarium" gebruikt de internationaal aanvaardde planeetsymbolen.
Na het kiezen van de routine krijgt U eerst een infoblok met een korte uitleg en de beschikbare commando's.
Dan vraagt de routine de startdatum en de startsnelheid (stapgrootte in aardse dagen).
Na de start ziet U rechts onder de datum, links onder de stapgrootte en bij "1" en "2" rechts boven de planeetsymbolen met een getal erachter. Dit is een "rondeteller" die aangeeft hoe vaak de betreffende planeet na de start het lentepunt in positieve zin (linksomdraaiend) is gepasseerd.

Tijdens het draaien van de routine kunt U m.b.v. de volgende toetsen:
0-9 :de stapgrootte veranderen
- : het teken van de stapgrootte veranderen (planeten lopen achteruit)
S : de planeetsymbolen laten afdrukken of weghalen (alleen bij "1" en "2")
B : de planeetbanen laten tekenen of weghalen (behalve "4")
spatie: kijken hoe de andere planeten staan op de datum rechts onder (alleen "1" en "2"). Bij routine "4" ziet U eerst de binnenste, dan de buitenste planeten.
? : informatieblok hehalen
M : terug naar het menu
H : routine herstarten, U kunt dan een nieuwe startdatum invullen

Vult U bij jaar ,maand of dag de letter M in, dan keert U terug naar het menu.
"5" Meteoorsporen
De enige niet-wetenschappelijke routine, U ziet een simulatie van een meteorenzwerm die verbrand in de dampkring tegen een willekeurige sterrenhemel
"6" Komeetbanen
Een interessante routine die de differentiaal-vergelijking uitrekent van een komeetbaan in het gravitatieveld van een zeer grote massa (bijvoorbeeld onze zon).
De routine vraagt de startpositie van de komeet (Rx en Ry) en de startsnelheid (Vx en Vy) (floating input).
Daarna vraagt ze de iteratie stapgrootte
Vervolgens ziet U:
-- de massa met assenkruis (voor het verduidelijkenn van de plaats van de kommeet)
-- momentele waarden van plaats en snelheid van de komeet (boven in beeld)
-- stapgrootte S (linksonder)
-- momentele waarde van het baan-impulsmoment L (rechtsonder)
-- de komeetbaan (na enige tijd)

Een paar leuke voorbeelden:

```
cirkelbaan: Rx=1      Vx=0
            Ry=0      Vy=1      S=5
ellipsbaan: Rx=0      Vx=.8
            Ry=-1     Vy=.75    S=5
parabool  : Rx=2.5    Vx=-1.5
            Ry=.5     Vy=0      S=5
```

U ziet zo een demonstratie van de wet van Keppler.
Per iteratiestap wordt een rekenfout gemmakt, waarvan U de invloed kunt zien opde waarde van het baanimpulsmoment. Deze behoort constant te blijven, dit wordt goed benaderd bij een kleine stapgrootte.
Raakt de komeet de massa, dan volgt een grote rekenfout (kijk maar naar de waarde van L), waarna de verdere baan niet meer correct is.
Tijdenns het draaien van de routine kunt U met:

```
 0-9  :de stapgrootte veranderen
 "?"  :informatie laten herhalen
 "M"  :terug naar het menu
 "H"  :routine herstarten.
```

Bij een floating point error, bijvoorbeeld het starten op de massa, klinkt een "bell" en kunt U opnieuw invoeren.

Tot slot
Het zal U duidelijk zijn dat "planetarium" ruim voorzien is van tekst en uitleg in het nederlands.
Het programma rekent intern met de zogenaamde Juliaanse tijd, en houdt dus rekening met schrikkeljaren.
De berekening van de planeetbanen is zeer nauwkeurig, dit geschiedt namelijk met inbegrip van tweede orde termen, de maan zelfs tot en met derde orde termen.
De relatieve traagheid van de sinusfunctie van de F.P.-rom is omzeild door gebruik te maken van een sinustabel (de F.P.-rom is uiteraard wel vereist!).
Bij routine "6" is gebruik gemaakt van de SET-routine

Opbouw en geheugengebruik:
Het programma draait geheel in GRMOD en heeft dus de josbox nodig.
Vanaf #2900 is versie 38.3 21327 bytes lang en heeft geheugen nodig tot #7FFF:

```
#2900-#2A00   startprogramma,buffer
#2A00-#3B00   shape-tabellen
              inputroutine
              sinustabel
              Set-routine
#3B00-#7C4F   hoofdprogramma (16719 bytes)
#7C60-#7FFF   array opslag
```

Verder is geheugen nodig van #1000 tot #2800 voor het weg- en terugcopieren van het scherm.
Het programma kan draaien zonder dit stuk geheugen, alleen mag dan nimmer het scherm verlaten anders dan door "M" (menu) of "H" (herstart)

We hopen dat U veel genoegen aan dit programma zult beleven, en dat Uw blik op ons zonnestelsel erdoor verruimd zal worden.

p.s. laat routines die iets tekenen lekker lang lopen!

NIEUWE WCH-ROUTINE

```
 10 REM NIEUWE WCHR
 20 REM AUTEUR: A.G.GERAETS
 30 REM         NEER
 40 REM DATUM : 840411
 50 REM PGM GEEFT FAST OUTPUT MET SINT MOGELIJKHEDEN
 60 REM DE SINT-ROUTINE IS ONTLEEND AAN EEN GELIJKNAMIG
 70 REM PGM UIT AN 1983
 80 DIM SS27;F.I=0TO27;SSI=#FFFF;N.
 90 IN."WAAR MOET ROUTINE KOMEN TE STAAN"Z
100 F.I=1TO2;P=Z;[
110:SS0;LDA@ SS1%256;STA#208;LDA@ SS1/256;STA#209;JMP#C55B
120:SS1;PHA;STX#E4;STY#E5;CMP@#0A;BNE SS2;JSR SS14
130:SS2;PLA;JSR #FEFB;PHP;PHA;CLD;JSR SS3;JMP #FE5F
140:SS3;CMP@#06;BEQ SS4;CMP@#15;BEQ SS5;LDY #E0;BMI SS6
150CMP @#1B;BEQ SS4;CMP @#07;BEQ SS7;JSR SS13;LDX @#0A
160JSR #FEC5;BNE SS10;JMP #FEB7
170:SS4;CLC;LDX @#00;STX #B000
180:SS5;LDX @#02;PHP;ASL #DE,X;PLP;ROR #DE,X
190:SS6;RTS
200:SS7;LDA @#05;TAY
210:SS8;STA #B003
220:SS9;DEX;BNE SS9;EOR @#01;INY;BPL SS8;RTS
230:SS10;CMP @#20;BCC SS13;ADC @#1F;BMI SS11;EOR @#60
240:SS11;STA(#DE),Y;INY;CPY @#20;BCC SS12;JSR SS19;LDY @#00
250:SS12;STY #E0
260:SS13;PHA;LDA (#DE),Y;EOR #E1;STA (#DE),Y;PLA;RTS
270:SS14;JSR #FE71;BCS SS18;CPY @#00;BEQ SS15
280CPY @#3B;JMP SS18
290:SS15;LDX @#20;JSR #FB83
300:SS16;JSR #FE71;BCS SS16;CPY @#00;BEQ SS17
310CPY @#3B;BEQ SS18
320:SS17;LDX @#10;JSR #FB83
330:SS18;RTS
340:SS19;LDA #DE;LDY#DF;CPY @#81;BCC SS26;CMP @#E0;BCC SS26
350LDY #E6;BMI SS22;DEY;BNE SS21
360:SS20;JSR #FE71;BCS SS20;LDY @#10
370:SS21;STY #E6
380:SS22;LDY @#20
390:SS23;LDA #8000,Y;STA #7FE0,Y;INY;BNE SS23
400:SS24;LDA #8100,Y;STA #80E0,Y;INY;BNE SS24;LDY @#1F;LDA @#20
410:SS25;STA (#DE),Y;DEY;BPL SS25;RTS
420:SS26;ADC @#20;STA #DE;BNE SS27;INC #DF
430:SS27;RTS
440]
450 P.#6;N.
460 @=0;P."ROUTINE AANROEPEN OP #"&SS0';@=8
470 END
```

DISK-KATALOGUS

Het programma DISKCAT wat gepubliceerd werd in Acorn Nieuws heb ik aangepast met de volgende mogelijkheden:

1. een overzicht van de files per disk
2. een snelle zoekroutine voor het vinden van files
3. een printprogramma voor de diskkatalogi
4. een sorteer- en printprogramma
5. een overzicht van alle disktitels
6. een programma om nieuwe files te updaten

Op de disk staan de volgende files:

DISKCAT: het hoofdprogramma
SETCAT : het updateprogramma
FCAT : het databestand (#3000 tot #67FF)
PCAT1 : dit programma maakt een random file van het databestand
PCAT2 : dit programma sorteert het databestand en print het uit
DFCAT : zoek- en printroutine (mach.taal),gebruikt door DISKCAT
PC-SORT: sorteerroutine voor PCAT2 (machinetaal)
PC-FILE: random-file voor PCAT
STARTDC: startprogramma dat de eerste keer de file FCAT maakt

DFCAT en PC-SORT zijn files in objectcode, maak deze files dus eerst aan met behulp van respectievelijk DFCTSRC en PCSOSRC.
In de programma's wordt gebruikt gemaakt van P-CHARME. Verder heb ik om de verschillende programma's in te laden het statement CHAIN gebruikt. Als U dit statement niet heeft kunt U dit vervangen door *RUN. U moet het te laden programma dan wel saven met BSAVE en zorgen dat de P-CHARME-box voorgeschakeld staat - met bijv. BEEP 20,30. Verder zal het duidelijk zijn dat er geheugen nodig is tot #6800 en voor het printprogramma PCAT zelfs tot #8000. Dit moet alleen als het totale databestand groter wordt dan ca. 750 files. Het grootste aantal files dat gesorteerd kan worden is 1130. Het maximale aantal schijven dat opgeslagen kan worden is 54.

FRANS LE BLANC

```
   10REM DISKCAT
   20REM VERSIE 2
   30*NOMON
   40*L.FCAT
   50*L.DFCAT
   60PAU.200;A=#28A0
   70R=0;F.N=#3105TO#67FFS.#108
   80R=R+?N/B;N.;@=2
   90P.$12;?#E1=0
  100P."               DISKCAT"'
  110P."               -------"''
  120P."   CATALOGUE        - KEY 1"'
  130P."   FIND FILES       - KEY 2"'
  140P."   PRINT CATALOGUE  - KEY 3"'
  150P."   PRINT INDEX      - KEY 4"'
  160P."   PRINT TITLES     - KEY 5"'
  170P."   SET NEW FILES    - KEY 6"'
  180P.'"   NUMBER OF FILES:  "R''
  190INKEYI
```

```
200XIF I>#30 A.I<#37
210ELSE BE.20,30;G.190
220IFI=#32;G.310
230IFI=#33;G.520
240IFI=#34;CHAIN"PCAT1"
250IFI=#35;G.630
260IFI=#36;CHAIN"SETCAT"
270IN."type disknumber "$A
280IF$A="";GOS.a;G.90
290IF VALA>54 OR VALA<=0;P.$11;LINK#FE22;G.270
300GOS.b;G.90
310IN."find "$A;IFL.A>7;P.$11;LINK#FE22;G.310
320IF$A="";G.90
330$#140=$A;P.$12"  "$A''
340F.N=1TO54;B=(N-1)*#108+#3000
350COPY B,B+#107,#2000;Q=?#2105+#2008
360F.I=#200F TO Q S.8;?I=#D;N.
370LINK#2810
380N.;P.'"ready  - PRESS KEY";BEEP20,5;LINK#FFE3;G.90
390?#E1=#80;E.
400aN=0;DO N=N+1
410GOS.c
420P."presscontrolforstop"
430GOS.d
440LINK#FFE3;U.N=54 OR?#B001=191;R.
450bN=VAL A
460GOS.c;P." ";GOS.d
470LINK#FFE3;R.
480cB=(N-1)*#108+#3000
490COPY B,B+#107,#2000
500LINK#2700;?#E1=0;R.
510dCU.11,0;P."  DISK "N;R.
520INPUT"BEGIN BIJ DISK:"B
530INPUT"EIND BIJ DISK: "E
540P.$2$27$64$27$48$27$67$0$12
550P.$27$67$80
560F.N=B TO E
570P."              DISK"N''
580C=(N-1)*#108+#3000
590COPY C,C+#107,#2000
600LINK#2705;P.''
610N.;P.$3
620G.90
630P.$12;?#E1=0;P=0
640F.N=1TO54
650C=(N-1)*#108+#3000
660COPY C,C+#107,#2000
670XIF?#2000<>32;P.N"  "
680F.J=#2000TO#2007
690XIF ?J<#20;P." "
700ELSE P.$(?J)
710N.
720F.J=#2100TO#2104
730XIF ?J<#20;P." "
740ELSE P.$(?J)
750N.;P.'
760ELSE
770N.;IF P=1;P.$3$6;G.90
780P.'"printer aan";INKEY I
```

```
790IF I=CH"J";P=1;P.$21$2$27$64;G.640
800G.90
```

```
 10PROGRAM SETCAT
 20DIMB1
 30DO P.$12"                  SETCAT"'
 40P."                  ------"''
 50P."INSERT DISK"'
 60VTAB8;P."PRESS CONTROL FOR STOP";CU.0,5
 70IN."TYPE DISKNUMBER"$B
 80IFVALB<=0 OR $B="" OR L.B>2;P.$11;LINK#FE22;G.70
 90A=VALB
100IFA>54;P."MAX.54 DISK'S!"'$11$11;LINK#FE22;G.70
110LINK#E223;Q=?#2105+#2008
120FILL Q,#20FF,0
130C=A*#108+#2EF8
140COPY#2000,#2107,C
150U.?#B001=191;P.$12"INSERT DISK FOR SAVING FCAT"';LINK#FFE3
160*N.
170*S.FCAT 3000 67FF
180P."FILE SAVED"''"READY"'
190CHAIN"DISKCAT"
```

```
 10 PROGRAM PRINTCAT-[PCAT1]
 20 P.$12;?#E1=0
 30 P."              PRINTCAT"'
 40 P."              --------"''
 50 P."EERST WORDT ER EEN FILE GEMAAKT"'
 60 P."EN DAARNA GESORTEERD."'
 70 P."ZET DE PRINTER VAST AAN..."'
 80 F.N=#3007 TO #67F8 S.8
 90 ?N=#D;N.
100 I=0
110 F=FOUT "PC-FILE"
120 F.N=1 TO 54
130 B=(N-1)*#108+#3000
140 Q=?(B+#105);C=B+Q
150 F.J=B+8 TO C S.8
160 IF C>B+8;I=I+1;SPUT F,$J
170 N.;SPUT F,"";PUT F,N
180 N.
190 SHUT F
200 CHAIN"PCAT2"
```

```
 10PROGRAM PRINTCAT-[PCAT2]
 20IF I>1120;P."TEVEEL FILES !!";PAU.150;G.400
 30ON ERROR;IF?0=30;P."WRITE PROTECT AAN";PAU.250;RUN
 40DIMTT(I),UU(I),V2
 50$V="  ";J=-1
 60F.N=0 TO I
 70DIMT7;TT(N)=T
 80DIMU2;UU(N)=U
 90N.
100F=FIN"PC-FILE"
110F.N=1 TO 54
120K=J+1
```

```
130DOJ=J+1
140SGETF,T;$TT(J)=$T
150U.$T="";J=J-1
160A=GETF
170F.M=K TO J
180?V=(A/10+#30);V?1=(A%10+#30)
190$UUM=$V
200N.;N.;SHUTF
210P.'"HET PROGRAMMA GAAT NU SORTEREN"''
220J=J+1
230?#80=TT0%256;?#81=TT0/256
240?#84=TTJ%256;?#85=TTJ/256
250?#86=TT(J-1)%256;?#87=TT(J-1)/256
260*R.PC-SORT
270P.$21
280P.$2$27$64$27$48
290P.$27$67$0$12
300P.$14$27$45$1
310F.N=1 TO 15;P." ";N.
320P."I N D E X"''''
330P.$27$45$0
340J=J-1;K=J/6+1
350F.N=0 TO K-1
360F.M=N TO N+K*5 S.K
370IFM<=J;P.$TTM"  "$UUM"  "
380N.;P.';N.;@=4;P.'J+1" files"'
390P.$12$3$6
400CHAIN"DISKCAT"
```

```
 10 REM DFCAT SOURCE
 20 REM SOURCE 2700-286B
 30 J=50;DIMLLJ;F.I=0TOJ;LLI=#FFF;N.;DIMA4;@=0
 40 P.$12" DFCAT SOURCE "
 50 P.''"CODE ASSEMBLEREN OP #"&#2700''
 60 P=#2700;Z=P
 80 LL1=#140;LL2=#32F;LL3=#2000;LL4=#2008;LL5=#200E
 90 LL6=#200F;LL7=#20F8;LL8=#2105;LL9=#2106;LL10=#2107
100 LL11=#210C;LL12=#210D;LL13=#210E;LL14=#210F;LL41=#C589
110 LL42=#E0EC;LL43=#E0FB;LL44=#E100;LL45=#E101;LL46=#E109
120 LL47=#F7D1;LL48=#F7FD;LL49=#FFED;LL50=#FFF4
130 P.$21;F.I=1TO2;P=Z
140[
150:LL0 LDA@#0C;JSRLL50;JSRLL49;LDX@#00;STX#B6
160:LL15 LDALL3,X;CPX@#08;BCCLL16;LDALL7,X
170:LL16 JSRLL50;INX;CPX@#0D;BNELL15
180:LL17 LDY@#00;JSRLL20;BCCLL27
190:LL18 JSRLL46;LDALL4,Y;AND@#7F;STALL4,Y;TYA
200 BNELL18;JSRLL49;JSRLL49;RTS
210:LL19 JSRLL44
220:LL20 CPYLL8;BCSLL21;LDALL4,Y;BMILL19
230:LL21 RTS
240:LL22 LDY#B8;CPY@#02;BNELL23;JSRLL49;LDY@#FF
250:LL23 INY;STY#B8;JSRLL32
260:LL24 LDA@#23;LDY#B7;LDXLL6,Y;BMILL25;LDA@#20
270:LL25 JSRLL50;LDX@#00
280:LL26 LDA#AE,X;JSRLL50;INX;CPX@#07;BNELL26
290 BEQLL17
300:LL27 STY#B7;LDX@#00
```

```
310:LL28 LDALL4,Y;AND@#7F;STA#AE,X;INY;INX
320 CPX@#08;BNELL28
330:LL29 JSRLL20;BCSLL31;LDX@#06;SEC
340:LL30 LDALL5,Y;SBC#AE,X;DEY;DEX;BPLLL30
350 JSRLL45;LDALL6,Y;AND@#7F;SBC#B5;BCCLL27
360 JSRLL44;BCSLL29
370:LL31 LDY#B7;LDALL4,Y;ORA@#80;STALL4,Y;LDA#B5
380 CMP#B6;BEQLL22;STA#B6;JSRLL49;LDA#B5
390 JSRLL50;LDA@#3A;JSRLL50;LDY@#00;STY#B8
400 BEQLL24;LDALL13,Y;JSRLL43;STA#A2;CLC
410 LDA@#FF;ADCLL11,Y;LDALL14,Y;ADCLL12,Y;STA#A3
420 LDALL13,Y;AND@#0F;ADC#A2;STA#A2;SEC
430 LDALL10,Y;SBC#A3;PHA;LDALL9,Y;AND@#0F
440 SBC#A2;TAX;LDA@#00;CMP#A0;PLA
450 SBC#A1;TXA;SBC@#00;RTS
460:LL32 LDY@#02
470:LL33 JSRLL42;DEY;BNELL33;RTS;\
480 LDX@#00;STX#43;STX#34;STX#25;DEX
490:LL34 LDY@#00
500:LL35 INX;CPXLL8;BEQLL40;LDALL1,Y;CMP@#0D
510 BEQLL37;CMPLL4,X;BEQLL36;CPY@#00;BEQLL35
520 DEX;JMPLL34
530:LL36 INY;BNELL35
540:LL37 TXA;AND@#F8;PHA;TAX
550:LL38 LDALL4,X;CMP@#0D;BEQLL39;JSRLL50;INX
560 BNELL38;TXA;PHA
570:LL39 JSRLL47
580];$P=" DISK ";P=P+L.P
590[
600 LDALL2;STA#16;JSRLL41;JSRLL48;PLA
610 CLC;ADC@#07;TAX;JMPLL34
620:LL40 RTS
630];N.;P.$6'"CODE VAN "&Z" TOT "&P'';E.
```

```
 10 REM PC-SORT SOURCE
 20 REM SOURCE 2800-2871
 30 J=8;DIMLLJ;F.I=0TOJ;LLI=#FFF;N.;DIMA4;@=0
 40 P.$12" PC-SORT SOURCE "
 50 P.''"ASSEMBLEREN OP "&#2800''
 60 P=#2800;Z=P
 80 LL1=#88
 90 P.$21;F.I=1TO2;P=Z
100[
110:LL0 LDA#80;CLC;ADC@#0B;STA#82;LDA#81
120 ADC@#00;STA#83
130:LL2 JSRLL3;LDA#82;CLC;ADC@#0B;STA#82
140 LDA#83;ADC@#00;STA#83;CMP#85;BNELL2
150 LDA#82;CMP#84;BNELL2;LDA#80;CLC
160 ADC@#0B;STA#80;LDA#81;ADC@#00;STA#81
170 CMP#87;BNELL0;LDA#80;CMP#86;BNELL0
180 RTS
190:LL3 LDY@#FF
200:LL4 INY;CPY@#07;BEQLL8;LDA(#80),Y;CMP(#82),Y
210 BEQLL4;BCCLL8;LDY@#00
220:LL5 LDA(#80),Y;STALL1,Y;INY;CPY@#0B;BNELL5
230 LDY@#00
240:LL6 LDA(#82),Y;STA(#80),Y;INY;CPY@#0B;BNELL6
250 LDY@#00
```

```
260:LL7 LDALL1,Y;STA(#82),Y;INY;CPY@#0B;BNELL7
270:LL8 RTS
280];N.;P.'$6'"CODE VAN "&Z" TOT "&P'';E.
```

```
 10REM START DISKCAT [STARTDC]
 20P.$12"WANNEER ER VOOR DE EERSTE KEER  EEN FCAT GEMAAKT WORDT"
 30P." MOET DIT PROGRAMMA GERUND WORDEN"''
 40P."DOE DE DISK IN DE DRIVE"'
 50FILL#3000,#67FF,0
 60F.N=#3000TO#67FFS.#108
 70!N=#20202020;N!4=#20202020
 80!(N+#100)=#20202020;N.
 90P.'"DRUK OP EEN TOETS"';INKEY I
100*S.FCAT 3000 67FF
110P.'"KLAAR"';END
```


TOP- EN DIMPONITER AUTOMATISCH GOEDZETTEN

```
 10 REM TOP TG /V1\
 20 REM TOP EN DIM-POINTER GOEDZETTEN IN EEN PROGRRAMMA
 30 REM AUTEUR A.G.GERAETS
 40 REM          NEER
 50 REM TEL. (04759) 2012
 60 REM DATUM : 830731
 70 REM GEBRUIK:
 80 REM - IN BEGIN PROGRAMMA LINKEN (LL0)
 90 REM - ALS STATEMENT (TOP), DAN LAATSTE RTS
100 REM    VERANDEREN IN 'JMP#C558'
110 REM HANDIG BIJ AUTOSTART PROGRAMMA'S
120 REM P (REGEL 150) VERANDEREN IN GEWENSTE ADRES
130 DIMLL5
140 F.X=1TO2;P.$21
150 P=#2800
160[:LL0 LDA#12;STA#E;LDY@#0;STY#D;DEY
170:LL1 INY;LDA(#D),Y;CMP@#D;BNELL1
180 JSR#CDBC;LDA(#D),Y;BMILL2;INY;BNELL1
190:LL2 INY;JSR#CDBC;LDA#D;STA#23;LDA#E;STA#24;RTS
200]
210 N.;P.$6
220 END
```

DE INHOUD VAN HET GERESERVEERDE GEHEUGEN.

Hier volgt nu een lijst van alle geheugenplaatsen welke voor de ATOM basicinterpreter, DOS-rom en MONITOR-rom gereserveerd zijn. De adressen voor de floating-point zijn globaal weergegeven. De lijst is uitgebreider dan die in ATOMWARE en is samengestelt op basis van het gehele uitvlooien van bovenvermelde drie ROMS.

De zero-page adressen #00 - #80 zijn gereserveerd voor de basic-interpreter.

```
 0       error nummer.
         lengte instruktie in assembler programma.
 1       lage byte regel nummer.
 2       hoge byte regel nummer.
 3       index Y-register statement uitvoering.
 4       basicstack pointer.
 5       lage byte adres statement uitvoering.
 6       hoge byte adres statement uitvoering.
 7       teller van karakters in print routine.
 8       random getal.
 9       random getal.
 A       random getal.
 B       random getal.
 C       random getal.
 D       lage byte top basic programma.
 E       hoge byte top basic programma.
 F       testbyte decimaal of hexadecimaal printen.
         testbyte juiste code in assembler matrix.
         file-handler van random acces files.
10       lage byte adres error-handler.
11       hoge byte adres error-handler.
12       tekst-pagina waarin basic programma start.
13       stackpointer do-loop's.
14       stackpointer go-sub's.
15       stackpointer for-next's.
16 - 22  lage byte basicstack.
23       lage byte dimpointer.
24       hoge byte dimpointer.
25 - 31  een na laagste byte basic stack.
27       testbyte voor negatief printen.
34 - 40  een na hoogste byte basic stack.
43 - 4F  hoogste byte basic stack.
45 - 4E  printen van een waarde.
52 - 55  wordt gebruikt voor verschillende zaken waaronder:
52       lage byte sprongadres zoekroutine.
         save karakter statement om met de tabel te vergelijken.
         lage byte adres naam file.
         lage byte lengte te plotten X-as.
52 - 55  buffer voor omrekenen naar hexadecimaal.
52 - 53  pointer om stringvariabelen en array's te vullen of te lezen.
53       hoge byte sprongadres zoekroutine.
         hoge byte adres naam file.
```

```
        hoge byte lengte te plotten X-as.
54      lage byte laad adres.
        lage byte lengte te plotten Y-as.
55      hoge byte laad adres.
        lage byte lengte te plotten Y-as.
56      lage byte executie adres = #B2.
        geeft aan of te plotten lijn X-as negatief is.
57      hoge byte executie adres = #C2.
        save adres basic label.
        lage byte testpointer bij plotten van een lijn.
58      lage byte start adres voor saven.
        lage byte pointer voor het opzoeken van een regelnummer.
        lage byte sprong adres bij goto of gosub.
        geeft aan of te plotten lijn Y-as negatief is.
59      hoge byte start adres voor saven.
        hoge byte pointer voor het opzoeken van een regelnummer.
        hoge byte sprong adres bij goto of gosub.
        hoge byte testpointer bij plotten van een lijn.
5A      lage byte eind-adres voor saven.
5A      lage byte plotadres X-coordinaat.
5B      hoge byte eind-adres voor saven.
5B      hoge byte plotadres X-coordinaat.
5C      lage byte plotadres Y-coordinaat.
5D      hoge byte plotadres Y-coordinaat.
5E      save Y-register op beginstatement bij interpretatie.
60 - 7F floating point werkgebied.
64      lage byte adresseringscode bij assembleren.
65      hoge byte adresseringscode bij assembleren.
66      opcode na het assembleren.
67      eerste operand.
68      tweede operand.
69      lage byte gecomprimeerde mnemonic.
6A      hoge byte gecomprimeerde mnemonic.
74 - 7F floating point vijfde byte basicstack.
```

De zeoro-page adressen #80 - #FF zijn gereserveerd voor het operating sytem.

```
80 - BF vrij bij Cassette Operating system.
80 - 99 vrij bij Diskette Operating system.
```

Het Diskette operating system gebruikt de zeropage van #9A t/m #FF. Aangezien er met de indeling van de DOS rekening is gehouden met de 40 koloms VDU kaart worden deze adressen hier ook bij vermeld, evenals de adressen voor de normale VDU routines uit monitorrom.

```
9A      lage byte adres naam file.
9B      hoge byte adres naam file.
9C      lage byte start adres.
9D      hoge byte start adres.
9E      lage byte executie adres.
9F      hoge byte executie adres.
A0      lage byte grootte van de file.
A1      hoge byte grootte van de file.
A2      hoge byte plaats op schijf.
A3      lage byte plaats op schijf.
A5 - AB naam van de file.
AC      De qualifier in gebruik.
```

AD	De opdrachtcode voor de controller.
AE - B6	naam file voor alphabetiseren catalog.
B7	save catalog index *CAT
B8	test byte voor linefeed in *CAT
B9	de filehandler bij *EXEC.
BA	de filehandler bij *SPOOL
BB	save lage byte schrijfroutine bij *SPOOL
BC	save hoge byte schrijfroutine bij *SPOOL
BD	save lage byte schrijfroutine bij *EXEC
BE	save hoge byte schrijfroutine bij *EXEC
BF	save index bij *EXEC en *SPOOL
C0	testbyte op geopende buffers.
C1	de te gebruiken buffer.
C2	de te gebruiken filehandler.
C3	index naam in disk catalog.
C4	index naam in buffer catalog.
C5	teller.
C6	save geheugen index.
C7	save in gebruik zijnde drive nummer.
C8	save in gebruik zijnde qualifier.
C9	lage byte start adres te saven of te laden sector.
CA	hoge byte start adres te laden of te saven sector.
CB	totaal aantal te saven sectoren.
CC	testbyte op meer dan 256 sectoren.
CD	de huidige qualifier.
CE	VDU bevat het aantal vrije regels.
CF	VDU lage byte adres regel in bewerking.
D0	VDU hoge byte adres regel in bewerking.
D1	VDU hoogste bit geeft aan ofscherm aan of uit staat.
D2	VDU lage byte adres eerst volgende te beschrijven plaats.
D3	VDU hoge byte gecorrigeerd adres eerstvolgende te beschrijven plaats.
D4	VDU hoge byte ongecorrigeerd adres eerstvolgende te beschrijven plaats
D5	lage byte non-request interupt adres.
D6	hoge byte non-request interupt adres.
DE	lage byte schermregel voor cursor positie.
DF	hoge byte schermregel voor cursor positie.
E0	cursor plaats op regel.
E1	cursor inhoud.
E2	lage byte sprong adres voor control code routine.
E3	hoge byte sprong adres voor control code routine.
E4	save adres X-register in schrijf routine.
E5	save adres Y-register in schrijf routine.
E6	test byte voor page mode.
E7	test byte voor locktoets.
E9	save index inputbuffer.
EA	lage byte print pointer #E016
EB	hoge byte print pointer #E016
EC	track nummer.
ED	sector nummer.
EE	drive nummer.
EF	testbyte MON/NOMON.
F0	aantal pogingen van de controller.
F1	aantal te laden sectoren.
F2 - FD	laadt of schrijfroutine DOS.
F6	lage byte save adres
F7	hoge byte save adres.
FE	karakter welke niet naar de printer wordt gezonden.
FF	save adres accu bij een interrupt.

Het Cassette Operating system gebruikt de adressen #C0 t/m #FF. Enkele adressen zijn hierbij vrij. Voor alle adressen wordt een separaat een opgave gedaan voor het saven en het schrijven dit wordt aangegeven door een S en een L. Indien geen L of S wordt aangegeven, betreft het geen COS routine, maar een Operating System routine.

```
C0 L: byte welke is geladen.
   S: byte welke wordt gesaved.
C1 S: bitteller.
C2 L: hoge byte teller voor controle op juiste header.
C3 S: save adres Y-register in put-byte routine.
   L: lage byte teller voor controle op juiste header.
      save adres Y-register in get-byte routine.
C4 L: meet byte voor lengte flank-overgang.
C5 S: vergelijkings byte voor testen op flank-overgang.
C9 S: lage byte adres naam file.
   L: lage byte adres naam file.
CA S: hoge byte adres naam file.
   L: hoge byte adres naam file.
CB S: lage byte reload adres.
   L: lage byte laad adres.
CC S: hoge byte reload adres.
   L: hoge byte laad adres.
CD S: lage byte executie adres.
   L: test byte of er een laad adres is opgegeven.
CE S: hoge byte executie adres.
   L: ckeck-sum byte.
CF S: grootte file.
D0 S: lage byte blok nummer.
   L: lage byte blok nummer.
D1 S: hoge byte blok nummer.
   L: hoge byte blok nummer.
D2 S: test byte voor:
        a. eerste blok file
        b. start en eindadres onjuist.
        c. is er nog een volgende blok.
D3 S: lage byte pointer met adres te saven blok.
D4 S: hoge byte pointer met te saven blok.
   L: lage byte laad adres catalog band.
D5 S: lage byte einde file.
   L: hoge byte laad adrs catalog band.
D6 S: hoge byte einde file.
   L: lage byte executie adres catalog band.
D7 L: hoge byte executieadres catalog band.
D8 L: grootte file catalog band.
D9 L: lage byte bloknummer catalog band.
DA L: hoge byte bloknummer catalog band.
DB    test byte catalog band. (zie ook bij #D2.
DC S: checksum byte.
   L: checksum byte.
DD L: testbyte *FLOAD.
E8    lage byte schrijf-pointer routine #F7D1.
E9    hoge byte schrijf pointer routine #F7D1.
EA    testbyte voor MON/NOMON.
EC S: save adres X-register in put-byte routine.
   L: save adres X-register in put-byte routine.
```

ED - FA L: naam file catalog band.

Dan volgt nu een opgave van het blok 0 - RAM welke niet tot de zero-page behoort. Het gebied van #100 t/m #1FF is door de fabrikant van de 6502 gereserveerd als microprocessor stack. De ontwerpers van de ATOM hebben deze stack gehalveerd ten behoeve van het input- en stringbuffer.

```
100 - 13F input buffer.
140 - 17F string buffer.
180 - 1FF micro processor stack.
```

Bij de volgende vector adressen wordt zowel de inhoud voor de COS als de DOS gegeven.

```
            COS   DOS  omschrijving.
200 - 201   XXXX  E87B NMI interrupt.
202 - 203   C9D8  C9D8 error handler.
204 - 205   A000  A000 interrupt handler.
206 - 207   F8EF  E3E5 interpreteer vector.
208 - 209   FE52  FE52 schrijfroutine.
20A - 20B   FE94  FE94 leesroutine
20C - 20D   F96E  E477 load file routine.
20E - 20F   FAE5  E613 save file routine.
210 - 211   C2AC  EA61 lees file routine.
212 - 213   C2AC  ECA0 store file routine.
214 - 215   FBEE  EAF0 get byte routine.
216 - 217   FC7C  EBBC put byte routine.
218 - 219   FC38  E953 message handler.
21A - 21B   C278  EB9E shut file routine.
21C - 23F   vrij voor eigen gebruik.
240 - 2B8   opslag adressen for next loops.
2B9 - 2CE   opslag adressen do until loops.
2CF - 2EA   opslag adressen gosub loops.
2EB - 320   opslag adressen array's.
321 - 38C   opslag adressen variabelen.
38D - 3C0   opslag adressen labels.
3C1         lage byte adres laatste plot point X-as.
3C2         hoge byte adres laatste plot point X-as.
3C3         lage byte adres laatste plot point Y-as.
3C4         hoge byte adres laatste plot point Y-as.
3C5         laagste byte floating-point variabele in FGET en FPUT.
3C6         een na laagste byte.
3C7         middelste byte.
3C8         een na hoogste byte.
3C9         hoogste byte floating-point variabele in FGET en FPUT.
3C5 - 3D8   floating-point voor wiskundige benaderingen.
3D9 - 3FC   vrij voor eigen gebruik.
3FD         kleur nummer ingesteld door het COULOUR statement.
3FE         lage byte adres plot routine.
3FF         hoge byte adres plot routine.
```

4 adressen welke bestemd zijn voor de disk-controller 8271.

```
A00      lezen :status register disk-controller.
    schrijven :commando register disk-controller.
A01      lezen :resultaat register.
    schrijven :parameter register.
A02 schrijven :reset register.
```

Geheugen welke gereserveerd is voor het disk operating system.

```
2000 - 2007 eerste acht letters disk-naam.
2008 - 21FF catalog van de file-namen.
2008 - 200E naam file
200F        qualifer van de file
            bit 7 geeft aan of de file gelockt is.
2010 - 20FF rest van de file-namen.
2100 - 2104 laatste 5 letters van de disknaam.
2105        aantal files op schijf.
2106        hoge nibble de OPTION van de schijf (alleen voor de teac-drive's)
            lage nibble hoge byte maximale capaciteit schijf.
2107        lage byte capaciteit schijf.
2108 - 21FF catalog van de adressen.
2108        lage byte start adres file.
2109        hoge byte laad adres file.
210A        lage byte executie adres.
210B        hoge byte executie adres.
210C        lage byte grootte file.
210D        hoge byte grootte file.
210E        hoge nibble file groter als 64K.
            lage nibble hoge byte plaats op schijf.
210F        lage byte plaats op schijf.
2110 - 21FF rest adressen catalog.
2200 - 22FF catalog van de random acces file buffers.
2300 - 23FF file-buffer 1.
2400 - 24FF file-buffer 2.
2500 - 25FF file-buffer 3.
2600 - 26FF file-buffer 4.
2700 - 27FF file-buffer 5.

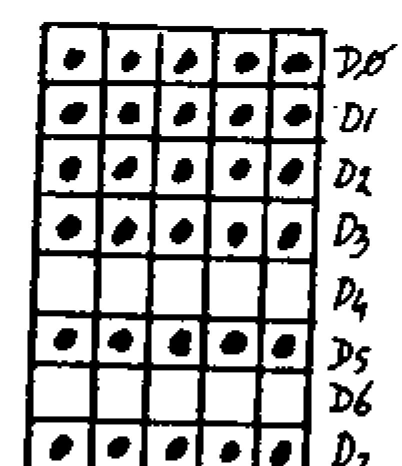
2800 - 2886 opslag adressen floating point variabelen.
2887 - 28bc opslag adressen floating point arrays.
```

SCREENDUMP 8023

```
 10 PROGRAM SCREENDUMP NEC PC-8023BE-N
 20 REM HENK BOXMA,OKT.'84
 25 REM GEBRUIK GEMAAKT VAN HET ARTIKEL VAN ANDRE DE BRUIN
 27 REM ZIE A.N.DEC'83,BLZ 35
 30 REM GEBRUIKT #9A T/M #9E ,ECHTER ALLEEN TIJDENS DE DUMP
 40 CLS
 50 P."SCREENDUMP OP NEC VAN CLEAR-4-  SCHERM.GEBRUIK CTRL-SHIFT"'
 60 W=#6000
 70 DIM VV9,TT3
 80 F.I=0 TO 9;VVI=W;N.
 90 F.I=0 TO 3;TTI=W;N.
100 P.$21
110 FOR I=1 TO 2;P=W
120  B=VV0%256;C=VV0/256
130  [
140  LDA@#40;STA#B80B;LDA@#C0;STA#B80E;LDA@#50;STA#B804
150  LDA@#FF;STA#B807;STA#B805;LDA@B;STA#204;LDA@C;STA#205
160  CLI;RTS
165
170  :VV0 LDA#B804;TYA;PHA;TXA;PHA
180  LDA#B001;CMP@63;\ CTRL SHIFT ?
190  BEQ VV7;JMP VV6
200  :VV7 LDA #B000;CMP@#F0
210  BEQ VV8;JMP VV6
220:TT0 ;];!P=#001B3E1B;P=P+3;$P="L005";P=P+LENP;?P=#1B;P=P+1
230 $P="T12";P=P+LENP;?P=#1B;P=P+1;$P="S0256";P=P+LENP
240 !P=#411B5D1B;P=P+4;REM TABEL OP TT0
245
250[ :VV8 LDX@0;STX#9A;LDA@#80;STA#9B;LDA@#2;JSR#FEFB
260  :TT1 LDA TT0,X;JSR VV9;INX;CPX@#B;BCC TT1
270  LDX@0;LDY@0
275
280:VV1 LDA@0;STA#9C;LDY@#B
290 :TT2 LDA TT0,Y;JSR VV9;INY;CPY @#11;BCC TT2
300  :VV5 LDA @#80;STA #9D
310   :VV4 LDY@0;STY#9E
320    :VV3 CLC;LDA(#9A),Y;AND#9D;BEQ VV2;SEC
330    :VV2 ROR #9E
340     TYA;ADC@#20;TAY;CMP@#C0;BNE VV3
350    LSR #9E;\ BIT 6 OP 0
360    SEC;ROR #9E;\ BIT 7 OP 1
370    LDA#9E;JSR VV9;\ PRINT BYTE
380    LSR #9D;BNE VV4
390   CLC;LDA#9A;ADC@1;STA#9A;LDA#9B;ADC@0;STA#9B
400   CLC;LDA#9C;ADC@1;STA#9C
410   CMP@#20;BNE VV5
420  LDA@#D;JSR VV9;\ NIEUWE REGEL
430  CLC;LDA#9A;ADC@#A0;STA#9A;LDA#9B;ADC@0;STA#9B
440  INX;CPX@#20;BNE VV1
450
460 LDY@#11
470 :TT3 LDA TT0,Y;JSR VV9;INY;CPY@#15;BCC TT3
480 LDA@#3;JSR#FEFB
490 :VV6;PLA;TAX;PLA;TAY
500 PLA;RTI
510 :VV9 PHA;JMP #FF08;\EEN VERKORTE ROUTINE VOOR PRINTEROUTPUT
520]
530 N.;P.$6
```

```
540 P."BEGIN CODE #"&W'
550 P."EINDE CODE #"&P'
560 P."PRESS KEY";INKEY T
570 LINK W
580 E.
```

<u>Toelichting:</u> In mode 4 wordt het geheugen van #8200 tot #9800 gebruikt.Het scherm is (al of niet) gevuld met puntjes:256 in de breedte en 192 in de hoogte.
We kunnen het scherm opgebouwd denken uit 192 rijen,elke rij bestaande uit 256 puntjes.
De eerste 8 puntjes van de bovenste rij vinden we terug in de 8 bits van het eerste byte,gerekend vanaf #8000.
Als het bitje een 0 is gebeurt er niets,als het bitje een 1 is,dan wordt een puntje op het scherm geplaatst.Dit is op de volgende wijze te demonstreren:
CLEAR4;FOR N=#8000 TO #9800;?N=#0F;NEXT
Dit levert een scherm vol met vertikale balken.Elke balk is 4 puntjes breed.
Sturen we de bytes,vanaf #8000, naar de printer dan mogen we verwachten dat er puntjes op het papier komen als de overeenkomstige bits 1 zijn.Om te beginnen moet de printer vantevoren geattendeerd worden op het feit dat het byte als een serie puntjes afgedrukt moet worden en NIET als een letter.Bovendien moet meegedeeld worden hoeveel bytes op die manier achter elkaar behandeld moeten worden.
Het commando P.#1B,"S0256" zorgt ervoor dat de NEC de eestvolgende 256 bytes in de "DOT-mode" zal afdrukken.
Hoewel een byte 8 bits bevat,zal de NEC slechts 7 bits daarvan gebruiken:bit0 t/m bit6.Het hoogste bit (bit7) <u>moet</u> 1 zijn, terwijl we dat bit NIET als een puntje op het papier terugvinden.(Een eigenschap van de *atom*)
Verzenden we byte #FF naar de NEC dan worden er dus 7 puntjes geprint.ECHTER niet horizontaal maar <u>vertikaal</u>,van boven naar beneden in de volgorde bit 0,bit 1,enz,t/m bit 6.
Welnu,dat houdt in dat we de schermhoogte van 192 puntjes zouden moeten verdelen in horizontale stroken van 7 bits.Aangezien 192 niet deelbaar is door 7,kunnen we het scherm beter verdelen in horizontale stroken van 6 puntjes.
Sturen we nu achter elkaar 256 bytes naar de printer en zorgen we ervoor dat elke byte een serie van 6 vertikale puntjes vertegenwoordigt,dan hebben we precies zo'n horizontale strook.
Voor elk byte moet nu dus gelden dat bit7=1 en bit6=0.
Nadat een horizontale strook is afgedrukt,moet de printkop terug naar het begin van de regel,en vervolgens moeten we ervoor zorgen dat de regelafstand gelijk is aan de hoogte van 6 vertikale puntjes.
Daarvoor wordt het commando P.#1B"T12" gebruikt.
De benodigde codes voor de printer zijn terug te vinden in de tabel TT0.
Het te printen byte wordt opgebouwd in #9E.Op regel 320 wordt m.b.v. #9D het juiste bit uit het zojuist gelezen byte "gepeuterd".In #9C wordt geteld hoeveel bytes reeds uitgelezen zijn.Als ?#9C=#1F dan zijn er 32*8=256 bits uitgelezen,en wel in 6 rijen "dik".



DATA: 5Keer achter elkaar #2F
(7e bit wordt genegeerd)

HARDWARE AANPASSINGEN

De Acorn Atom kan m.b.v. een klein aantal onderdelen vrij eenvoudig op een 3-tal punten worden verbeterd of aangepast.
Het betreft een hardware aanpassing zonder aan de hoofdprint in de Atom iets te moeten wijzigen.
De aanpassingen zijn:

1. onderdrukken van de soms irriterende sneeuwruis op het beeldscherm.
2. vervangen van het videogeheugen door een 8Kx8 CMOS Ram en dus tevens uitbreiding van het geheugen met 2K vanaf 9800.
3. het vrijmaken van een aantal moeilijk verkrijgbare IC's, die op een andere plaats in de Atom weer gebruikt kunnen worden.

1. Onderdrukken van de ruis.

De ruis op het beeldscherm wordt veroorzaakt door het schrijven of lezen in het videogeheugen door de CPU.
Deze ruis is soms zeer irriterend, vooral bij het runnen van programma's met graphics waarin shape tabellen voorkomen.
Deze ruis kan men onderdrukken m.b.v. 2 IC's, n.l. een one-shot (74LS123) en een 8-bits latch (74LS373).
De 74LS123 zorgt voor verlenging van de VDG-puls, zodat de data naar de video-generator vastgehouden wordt zodra er door de CPU in het videogeheugen geschreven of gelezen wordt.
Het onderdrukken van de ruis werkt echter alleen goed als het videogeheugen is uitgevoerd met snellere geheugens die een access-tijd hebben van 200 nsec. of kleiner.
Hiermee komen we tevens op het volgende punt.

2. Vervangen van het videogeheugen door een 8Kx8 CMOS Ram.

Door het videogeheugen (12 stuks 2114) te vervangen door een 8Kx8 CMOS ram kan men aan de eis van punt 1 gaan voldoen, n.l. access-tijd kleiner dan 200 nsec.
Dit is echter niet alles, want men krijgt gelijktijdig een uitbreiding van het geheugen met 2K vanaf adres 9800 t/m 9FFF.
Een andere gunstige bijkomstigheid is, dat de voedingsstroom ca. 0,5 amp. lager wordt.
Bovendien kan IC30 verwijderd worden omdat de adres-decoder nu overbodig wordt.

3. Vrijmaken van 3 moeilijk verkrijgbare IC's.

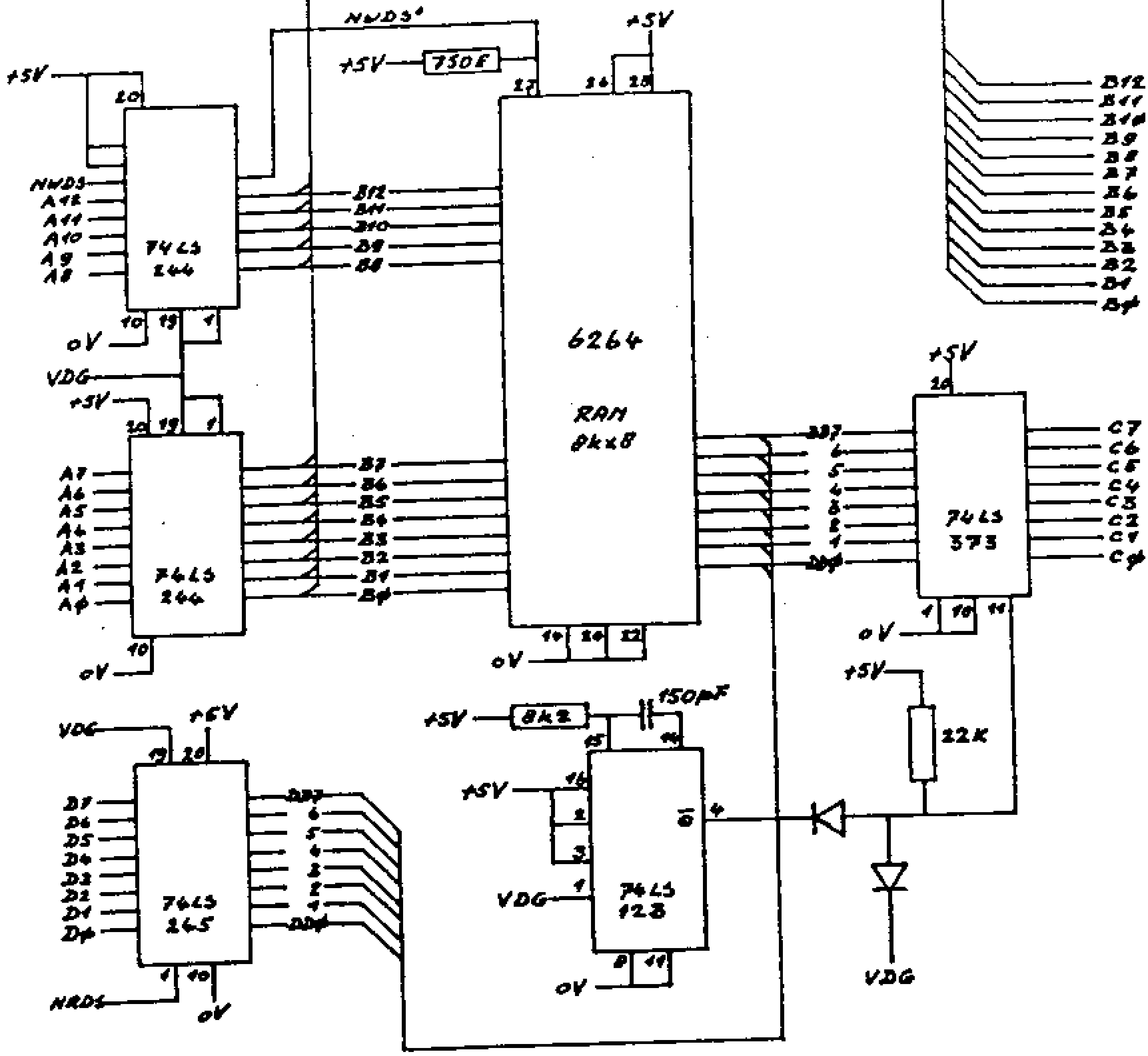
Voor uitbreiding van de Atom via de connectoren PL6/7 moeten IC 2,3,4 en 5 aangebracht worden. Echter de IC's 2 èn 3 (81LS95) en IC4 (8304) zijn nogal moeilijk verkrijgbaar, maar zijn wel te vervangen door de 74LS244 voor de 81LS95 en de 74LS245 voor de 8304, echter met het nadeel dat de aansluitingen van de IC's niet aan elkaar gelijk zijn, zodat men op het hoofdpaneel wijzigingen moet aanbrengen, n.l. sporen onderbreken en nieuwe verbindingen aanbrengen.
Dit probleem kan men oplossen door IC27,28 en 29 te verwijderen en deze te plaatsen in de voetjes voor de IC's 2,3 en 4.
Voor IC27,28 en 29 kan men nu de 74LS244 en de 74LS245 gaan gebruiken en deze samen met de aanpassingen uit punt 1 en 2 op een apart printje gaan monteren.

Praktische oplossing.

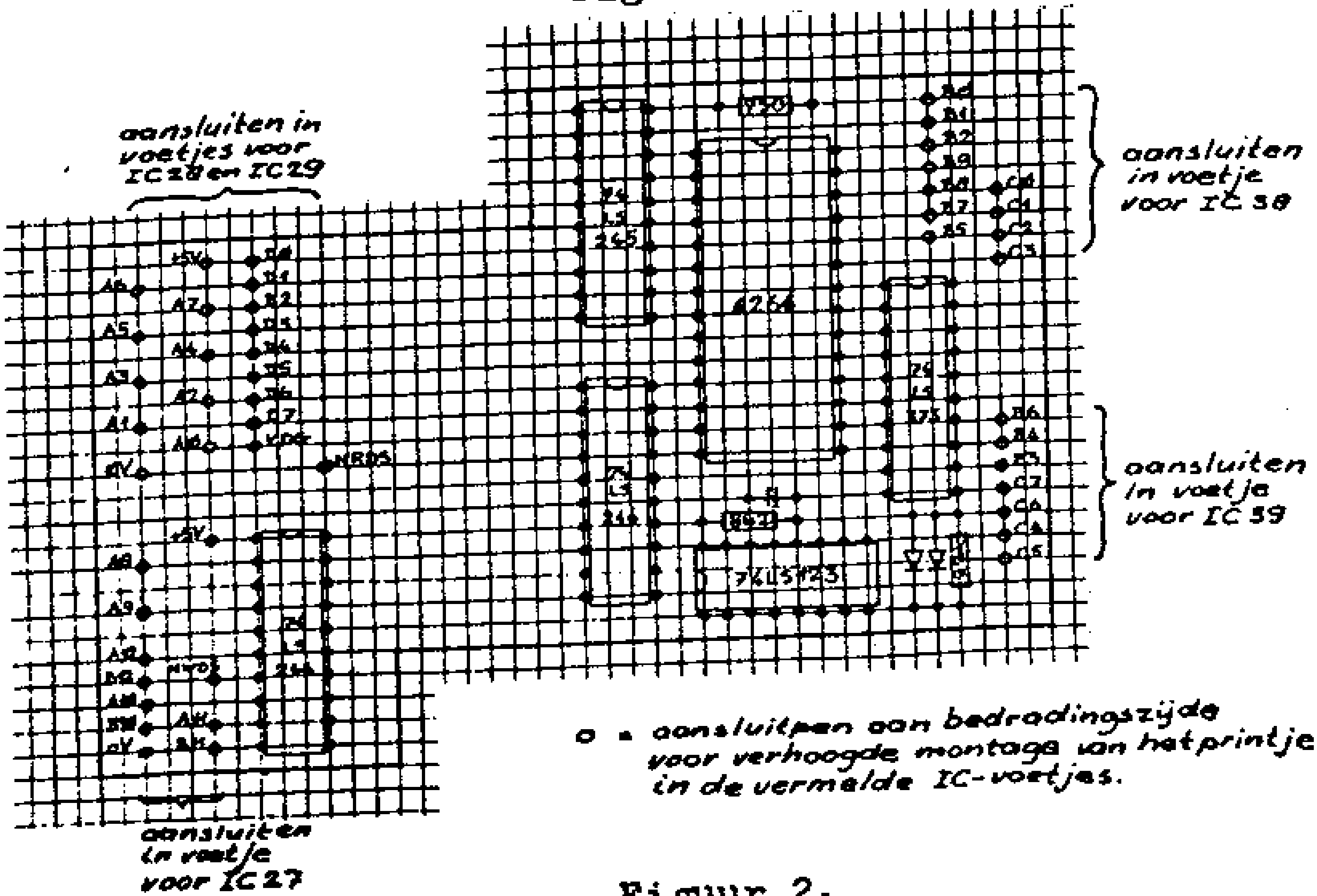
Het totale schema voor de aanpassingen uit de punten 1,2 en 3 ziet er uit volgens figuur 1.
Figuur 2 laat het printje zien waarop het geheel is gemonteerd (door mij op gaatjesprint gemaakt).
De IC's 27 t/m 30 en 32 t/m 43 moeten van de hoofdprint worden verwijderd, zodat het uitbreidingsprintje m.b.v. de aansluitpennen aan de bedradingszijde van het printje, verhoogd in een aantal van de vrijgekomen IC-voetjes geplaatst kan worden.
De open plek op het uitbreidingsprintje komt dan boven IC31.
De vorm van het printje als in fig. 2 is zo gekozen i.v.m. de bereikbaarheid van andere componenten op de hoofdprint.
Het stroomverbruik van de Atom kan nog verder beperkt worden door het toepassen van high-speed-cmos circuits i.p.v. de aangegeven 74LS... circuits.

Met het onderstaande programma kan de werking van de ruisonderdrukker getest worden.
Het programma loopt in het videogeheugen.

```
10 ?#E1=0; REM TURN OFF CURSOR
20 L=#9000; P=L
30 [LDA@0; BEQ L;] ; REM MACHINE CODE LOOP
40 CLEAR 0; REM TO CLEAR RAM ACCESSED BY VDG
50 LINK L; REM LOOP IN SCREEN RAM
60 END
```



Figuur 1.



Figuur 2.

STEM VOOR DE ATOM

Alhoewel er in diverse regiobladen al vaker over voice-synthesizers is geschreven wil ik er toch in ons regioblad nog eens een paar kantjes aan wagen. De aanleiding hiertoe is dat ik in het bezit ben gekomen van de voice-synthesizer SP0256-AL2 van General Instruments, welke door de Tandy winkels wordt verkocht.
Dit circuit heeft voor de hobbyist een paar interessante eigenschappen:
1) Het is goed verkrijgbaar, en betaalbaar!!
2) Het is eenvoudig op de ATOM aan te sluiten.
3) Er zit een goede en uitgebreide dokumentatie bij (engels!).
4) Alle engelse woorden kunnen worden uitgesproken.
Er is echter ook een nadeel:
5) Alhoewel de spraak goed verstaanbaar is, klinkt deze erg monotoon en "robot-achtig".
Bovengenoemde eigenschappen maken het circuit interessant voor die hobbyisten die wel eens met voice-synthesizing willen experimenteren, maar hiervoor geen honderden guldens over hebben.
Voor deze mede-atomisten wil ik de voorgaande punten nog eens wat verder uitspitten.

ALGEMEEN
ad.1)
Het circuit wordt verkocht door de Tandy winkels onder Cat.No: 276-1784 en het juiste type is: SP0256-AL2. De prijs is fl.54,-. Pas op, er bestaat ook een SP0256-017! Dit is een sprekende klok circuit dat alleen tijd en datum kan uitspreken! Hieraan heb je dus niets! (Ik spreek uit ervaring!).
Volgens de dokumentatie hoort er een kristal van 3.12MHz bij. Dit is een niet-standaard waarde, dus moeilijk verkrijgbaar en duur! Heel goed gaat het echter ook met en z.g.n. Color-Burst kristal van 3.579MHz. Dit is ook bij Tandy te koop en kost ca. fl.6,-. Voor zo'n fl.60,- heb je dus vrijwel alles, op wat weerstanjes condensatoren en draad na! (zie schema).
ad.2)
Het circuit kan direkt op poort B van de via 6522 worden aangesloten (zie schema). Dit circuit moet dan wel in de Atom aanwezig zijn! Met de VIA kan ook de synchronisatie (handshaking) goed worden afgewerkt door de lijnen CB1 en CB2. Het L.F.signaal over de 10K weerstand is niet erg groot, zodat een versterker met een gevoelige ingang nodig is (bijv. microfoon). Voor de eerste experimenten kan ook een kristal oortelefoontje worden aangesloten.
De benodigde programmatuur wordt verderop besproken.
ad.3)
In de verpakking bij de chip zit een boekje van 18 blz. waarin naast de technische gegevens ook een uiteenzetting over de wijze van spraak synthese zit, namelijk met z.g.n. Allophonen. Wat dit voor dingen zijn komt verderop ter sprake onder het hoofdstuk: "Hoe werkt het?". Overigens komt mijn wijsheid allemaal uit het bovengenoemde boekje! Tevens bevat het boekje een redelijke hoeveelheid voorbeelden. Nadeel is wel dat alles in het engels is geschreven. Ook de spraak is gebaseerd op het engels! Een aantal nederlandse klanken is niet aanwezig (de G bijv.!).
ad.4)
Om dit en punt 5) toe te lichten kan ik beter meteen naar het volgende hoofdstuk gaan, daar gaat ie dan!

HOE WERKT HET?
Uitgangspunt van deze vorm van spraak synthese is dat spraak kan worden opgebouwd uit een beperkt aantal klanken. Deze klanken hebben niet altijd een direkte relatie met de geschreven tekst, en verschillen bovendien afhankelijk

van de plaats waarin ze in een woord voorkomen. In het woord TEST bijv. wordt de T in het begin anders uitgesproken als die op het eind. Zo zijn er vele regels die de uitspraak van een woord bepalen. Bovendien is dit voor iedere taal verschillend.
Toch blijkt het voor een taal mogelijk om met een beperkt aantal klanken (allophonen) een redelijk intelligente spraak te synthetiseren. De SP0256-AL2 kent 59 van deze allophonen en 5 verschillende pauze lengten. Vooral deze laatste zijn erg belangrijk voor de verstaanbaarheid.
Door nu deze 59 voorgeprogrammeerde klanken samen met de 5 pauzen op de juiste wijze achter elkaar te laten klinken, kan verstaanbare spraak worden gegenereerd.
Het voordeel is dat er relatief weinig geheugen nodig is (ca. 10 bytes/seconde). De nadelen zitten in het beperkt aantal klanken. Hierdoor wordt de spraak erg monotoon, en de klanken zijn op een enkele taal gebaseerd. In dit geval engels dus!
De verschillende allophonen worden in het boekje door lettercombinaties weergegeven. Een soort mnemonics dus, net als bij assembler programmeren. Even een voorbeeld:
ACORN ATOM wordt in allophonen als volgt geschreven:
EY PA3 KK3 OR NN1 PA3 EH EH PA4 TT2 UH MM
Iedere lettercombinatie stelt dus een klank voor, de letters PA staan voor de verschillende pauzen. Uit het feit dat er NN1 en TT2 staat blijkt al dat er meerdere N-klanken en T-klanken zijn.
Het boekje bevat een overzicht van de regels die bepalen welke klanken waar thuishoren, met voorbeeldwoorden er bij.
In een tabel wordt van iedere allophone het adress wat op de databus moet worden gezet (een getal tussen 0 en 63) vermeld.
Het selekteren van een allophone vanuit de Atom gaat erg eenvoudig:

a) We zetten een getal tussen 0 en 63 op de databus (poort B).
b) We maken het signaal ALD (adress load) laag, en de chip begint te kletsen.
c) Het signaal LRQ (load request) vertelt ons wanneer de chip nieuwe data wil zien.
d) Indien nodig kan men aan SBY (standby) zien wanneer de chip is uitgepraat (wordt door mij niet gebruikt).

DE SOFTWARE

Achter dit verhaal zitten twee listingen:
Een basic programma waar m.b.v het READ en DATA statement de uit te spreken allophonen als een reeks getallen (adressen) in een DATA statement kunnen worden geplaatst. Hierin is ook duidelijk het hele output gebeuren naar poort B te zien, alsmede het lezen van LRQ via CB1 en het IRF-register van de 6522.
De tweede listing is die van een nieuw P-CHARME statement, welke ik SAY heb genoemd. Na assembleren van dit statement en koppeling aan P-CHARME is het mogelijk om na SAY de gewenste allophonen in de symbolische code, zoals deze in de datasheet staat, in te voeren. Een soort spraakversie van het PRINT statement dus.
Het uitspreken van ACORN ATOM gaat dan als volgt:
SAY EY PA3 KK3 OR NN1 PA3 EH EH PA4 TT2 UH MM PA4 <CR>
De spaties hebben geen betekenis en het interpreteren van de allophonen gaat door tot een <;> of een <CR>. Een onbekende letter combinatie genereert een foutmelding.
Even een opmerking: een woord moet altijd met een pauze eindigen, anders blijft de laatste klank klinken!
Met dit statement is het eenvoudig om de Atom hele verhalen te laten vertellen. Mijn ervaring is wel dat je beter woord voor woord kunt proberen door ze in een klein programmatje in een loop te zetten. Bijvoorbeeld:

```
10 INKEY A
20 SAY TT2 EH SS TT1 PA1
```

```
30 GOTO 10
```

OP Deze wijze is het simpel om een woord te perfectioneren.
Zeer belangrijk voor de uitspraak blijkt ook het op het juiste punt inlassen van pauzes. Het boekje geeft ook hiervoor enige richtlijnen. Van de goed geslaagde woorden kan men voor zichzelf een soort woordenboek aanleggen, waardoor het steeds eenvoudiger wordt de juiste allophonen te vinden.

TENSLOTTE

Wie helemaal door de spraak-koorts is gegrepen moet het drukwerk archief eens doorsnuffelen. In verschillende regiobladen staan namelijk ook artikelen over de SP0256.
Ook een kopie van het genoemde boekje/datasheet is in het archief aanwezig.
Met de twee niet gebruikte datalijnen (B6 en B7 zijn niet nodig voor 64 adressen!) kan eventueel een eenvoudige volume regeling worden gemaakt, waardoor sommige klanken meer nadruk kunnen krijgen. Hiermee kan het monotone karakter van de spraak wat verminderd worden. Helemaal mooi zou zijn als de frekwentie wat veranderd zou kunnen worden onder software besturing. Dan zou ook een vragende of bevestigende zin gemaakt kunnen worden. Maar omdat de frekwentie kristal gestuurd is, gaat dit niet erg eenvoudig. Mogelijk is het echter wel!
Zijn er vragen ,op- of aanmerkingen of komt men er helemaal niet meer uit dan ben ik altijd bereid met raad en/of daad terzijde te staan. Voor zover ik er zelf tenminste iets van begrijp!
Succes met de kletsende Atom!

DE LISTINGS

```
 10 PROGRAM TALK
 20
 30 PROC INIT
 40 ?#20B=#55
 50 ?#B80C=#EC
 60 ?#B802=#FF
 70 PEND
 80
 90 PROC OUT(A)
100 ?#B800=A
110 ?#B80C=#CC
120 ?#B80C=#EC
130 IF (?#B80D)&#10=0;G.130
140 PEND
150
160 REM HOOFDPROGRAMMA
170 INIT
180 READ D
190 IF D>63;END
200 OUT(D)
210 GOTO 180
220
230 DATA 6,4
240 DATA 7,16,4
250 DATA 18,19,4
260 DATA 20,2,8,58,11,2
270 DATA 7,7,3,13,30,16,4
280 DATA 64
```

```
 10 PROGRAM SAY.STAT V0
 20
 30 DIM S6,LL6,TT0
 40 F.I=0TO6;LL6=A;N.;TT0=A
 50 P.$21;P=A;GOS.a
 60 P.$6;P=A;GOS.a
 70 RESTORE
 80 FOR N=0 TO 63
 90 READ $S
100 $P=$S;P=P+LENP+1;?(P-1)=#FF
110 NEXT
120 ?(P-1)=#EE
130 $T="SAY";T=T+LENT
140 ?T=LL0/256|#80;T?1=LL0%256;T?2=#80;T
[illegible] A=P;T!1=A
[illegible] P."KLAAR TOP="&P
[illegible] END
[illegible]
```

```
190a[
200:LL0 LDX@#FF START VALUE X POINTER
210     STX#90 INITIALIZE ALLOPHONE COUNTER
220:LL1 LDY#03 LOAD CHAR.POINTER
230     INC#90 INCREMENT ALLOPHONE COUNTER
240     DEY
250:LL2 INY INCREMENT CHAR.POINTER
260     LDA(#05),Y UNTIL CHAR. IS
270     CMP@#20 NOT A SPACE
280     BEQLL2
290     DEY
300:LL3 INX
310     INY
320     LDATT0,X LOAD TAB.CHAR
330     BMILL5 IF B7=1:STRING RECOGNIZED!
340     CMP(#05),Y COMPARE TABCHAR-RAMCHAR
350     BEQLL3 IF EQUAL:NEXT CHAR
360     DEX
370:LL4 INX IF NOT EQUAL:
380     LDATT0,X SEARCH NEXT
390     BPLLL4  TABSTRING.
400     CMP@#EE  END OF TABEL?
410     BNELL1 IF NOT:NEXT TABSTRING
420     STY#03 STORE CHAR.POINTER
430     JMP#C31B BACK TO BASIC
440:LL5 STY#03 STORE CHAR.POINTER
450     LDA@#55
460     STA#208 REMOVE PRINTER
470     LDA@#EC
480     STA#B80C CB2:=1
490     LDA@#FF
500     STA#B802 DDRB:=OUTPUT
510     LDA#90 LOAD ALLOPHONE
520     STA#B800 OUTPUT
530     LDA@#CC
540     STA#B80C CB2:=0
550     LDA@#EC
560     STA#B80C CB2:=1
570:LL6 LDA#B80D LOAD IFR
580     AND@#10 MASK BITS
590     BEQLL6 WAIT TILL B4=1
600     LDA@#00
610     STA#B80C RESET PCR
620     LDA@#52
630     STA#208 RESTORE PRINTER
640     JMPLL0 NEXT ALLOPHONE
650:TT0;];
660 RETURN
670 DATA"PA1","PA2","PA3","PA4","PA5","OY","AY","EH","KK3"
680 DATA"PP","JH","NN1","IH","TT2","RR1","AX","MM","TT1"
690 DATA"DH1","IY","EY","DD1","UW1","AO","AA","YY2","AE"
700 DATA"HH1","BB1","TH","UH","UW2","AW","DD2","GG3","VV"
710 DATA"GG1","SH","ZH","RR2","FF","KK2","KK1","ZZ","NG"
720 DATA"LL","WW","XR","WH","YY1","CH","ER1","ER2","OW"
730 DATA"DH2","SS","NN2","HH2","OR","AR","YR","GG2","EL"
740 DATA"BB2"
```

HET SCHEMA STAAT OP PAG. 15

INTIKKEN EN RUNNEN MAAR

DIKTEE

```
10REM DIKTEE
20P.$12;P."DIT IS EEN DIKTEE OEFENING"';GOS.v
30P."DE NA TE TYPEN WOORDEN BLIJVEN   3 SEC ZICHTBAAR"''
40 GOS.v
50@=0
60 J=0
70 IN."WELKE LES:"'"TYPE IN 1   2  OF 3 "''L;L=L+96
80 IN."HOEVEEL OPGAVEN "'"TYPE  IN GETAL TUSSEN 5 EN 10"''Z
90 P.'''"EVEN GEDULD "
100 DIM A(40);S=0;DIM X(6)
110 GOS.f
120 IF J=Z G.240
130 GOS.r
140R=0;GOS.v;P.$12;P.'''''''
150P."TYPE NA  *"$A"*";GOS.v;P.$12
160 P.''''''''"...";IN.$L
170 IF$L=$A P."GOED"'"VOLGENDE WOORD   "''';S=S+1;G.120
180P."FOUT"'"OPNIEUW"'';R=R+1
190 IF R=3 P."HET MOET * "$A" * ZIJN"'';GOS.v;G.230
200G.150
210 E.
220vF.N=1TO150 ;WAIT;N.N;P.';R.
230 P."   DRUK OP EEN TOETS*"';LINK #FFE3;P.$12;G.120
240 P.  "JE HEBT VAN DE  "J" OPGAVEN              ER  "S" GOED"''
250GOS.v;S=0;J=0
260P. "WIL JE NAAR EEN ANDERE  LES     OF MEER OPGAVEN"''
270 IN."TYPE DAN *JA* ZONIET DRUK RETURN"$X
280 IF $X="JA";G.60
290 G.120
300fREM POINT Q TO TEXT
310 Q=?18*256
320 DO Q=Q+1
330 U.?Q=#D AND Q?3=L
340 Q=Q+4 ;R.
350*
360rREM READ NEXT ENTRY INTO A
370 J=J+1
380 REM CHANGES:A,Q,R
390 R=-1
400 DO R=R+1;A?R=Q?R
410 U.A?R=CH","ORA?R=#D
420IF A?R=#D Q=Q+3
430Q=Q+R+1;A?R=#D;R.
440*
450aIN,UIT,OP,AF,OP,MAN,MOGELIJKHEDEN
460 PROTESTEN,KWART,GEWOONTEN,HOEVEELHEDEN,HONDERDDUIZEND
470 BEDRIJVEN,HUWELIJKEN,INDIANEN,ZEELAND,HERINNERINGEN
480bNEDERLANDER,FEITEN,HOLLAND,BEWEGINGEN,LEERLINGEN,DIENSTEN
490 DE FRIEZEN SPREKEN FRIES,BOVEN IN JE HOOFD ZITTEN HERSENEN
500 EEN FRANSMAN SPREEKT FRANS
```

```
510cBLAUW,DOKTER,MORGEN,ZOLDER,FLAUW,ALLE,ALLEMAAL,GAUW,PAUS
520 BAKKER,ANDERS,PAUW,WALVIS,ELKAAR,SAUS,WORTEL,JULLIE,MEESTER
530 NAUW,HUISVROUW,ALLES,KELDER,OUDER,ZOUT,TANTE,KASTEEL,BOUW
540 OUD,MANTEL,GULDEN,HOUT,JOU,MEESTE,ONDER,KABOUTER
550 ZAKDOEK,ZESDE,FILM,FLES,ZIEKE,ZWAAN,FLUIT,ZOETE,ZWAAR,FRAAI
560 FRAAI,ZONDER,ZWEET,FRIS,ZUSTER,ZWAK,ZOON,ZWARTE,FRUIT,ZAAK
```

VERTAALPROGRAMMA

```
10REM VERTAAL PROG.
20P.$12;P."DIT IS EEN VERTAAL OEFENING"'
30P."VERTAAL DE VOLGENDE WOORDEN OF   ZINNEN:"''
40P."IN HET ENGELS *1*"''
50P."IN HET DUITS  *2*"''
60P."IN HET FRANS  *3*"''
70@=0;S=0
80 IN."WELKE TAAL?"'"TYPE IN 1   2  OF 3 "''L;L=L+96
90 IN."HOEVEEL OPGAVEN "'"TYPE  IN GETAL TUSSEN 5 EN 10"''Z
100 P.''''"EVEN GEDULD "
110 DIM A(40);S=0;DIM X(6);DIM P(40)
120 GOS.f
130GOS.v
140 IF U=Z G.260
150 GOS.r
160R=0;P.$12;P.'''''
170P."VERTAAL  *"$A"*"
180 P.'''     "...";IN.$L
190 U=U+1
200 GOS.r;IF $A=$L P."GOEDZO"''"VOLGENDE WOORD";S=S+1;G.130
210P."FOUT HET MOET ZIJN"''$A
220 G.130
230 E.
240vF.N=1TO150 ;WAIT;N.N;P.';R.
250 P."   DRUK OP EEN TOETS*"';LINK #FFE3;P.$12;G.130
260 P.''"JE HEBT VAN DE  "U" OPGAVEN     ER "S" GOED"''
270 P."JE CIJFER IS EEN ";P.10*S/U ;P.''
280GOS.v;S=0;U=0
290P. "WIL JE NAAR EEN ANDERE  LES     OF MEER OPGAVEN"''
300 IN."TYPE DAN *JA* ZONIET DRUK RETURN"$X
310 IF $X="JA";G.70
320 G.130
330fREM POINT Q TO TEXT
340 Q=?18*256
350 DO Q=Q+1
360 U.?Q=#D AND Q?3=L
370 Q=Q+4 ;R.
380*
390rREM READ NEXT ENTRY INTO A
400 J=J+1
410 REM CHANGES:A,Q,R
420 R=-1
430 DO R=R+1;A?R=Q?R
440 U.A?R=CH","ORA?R=#D
450IF A?R=#D Q=Q+3
460Q=Q+R+1:A?R=#D:R.
```

```
470*
480aDEUR,DOOR,HUIS,HOUSE,BOOM,TREE,KAT,CAT,OUD,OLD
490 WACHT,WAIT,PAPIER,PAPER
500bKAMER,ZIMMER,BOOM,BAUM,ZEE,MEER,MEER,SEE,TREIN,ZUG
510cBANK,FAUTEUILE,MAN,HOMME
```

CIRCLES

```
 10 PROGRAM CIRCLES
 20
 30 REM (C) HAMBURGERSOFT
 40 REM UIT CREATIVE GRAPHICS
 50
 60 PROC CIRCLES(X,Y,%R,L,P)
 70  IF L=5 THEN PEND
 80  %A=0
 90  DO
100   MOVE X,Y
110   %X=%R*COS%A
120   %Y=%R*SIN%A
130   DRAW(%X+X),(%Y+Y)
140   %A=%A+PI/%R
150  FU.%A>=2*PI-PI/%R
160  IFP<>3;CIRCLES((X-%R),(Y-%R),(%R/2),(L+1),1)
170  IFP<>4;CIRCLES((X+%R),(Y-%R),(%R/2),(L+1),2)
180  IFP<>1;CIRCLES((X+%R),(Y+%R),(%R/2),(L+1),3)
190  IFP<>2;CIRCLES((X-%R),(Y+%R),(%R/2),(L+1),4)
200 PEND
210
220 CLEAR4
230 CIRCLES(128,96,48,0,0)
240 DO;UNTIL 0
```

BOX

```
 10 PROGRAM BOX
 20
 30 REM (C) HAMBURGERSOFT
 40 REM TUMBLING BOX
 50 REM UIT CREATIVE GRAPHICS
 60 REM VOOR DE BBC MICRO
 70
 80
 90 PROC BOX(P,A)
100  %X=%XXA;%Y=%YYA
110  MOVE(%X+128),((A/5)+96)
120  PLOT P,-%X,-%Y;PLOT P,-%X,%Y;PLOT P,%X,%Y;PLOT P,%X,-%Y
130  PLOT P,%Y,%X
140  PLOT P,-%X,-%Y;PLOT P,-%Y,-%X;PLOT 0,%Y,%X
150  PLOT P,-%X,%Y;PLOT P,-%Y,-%X;PLOT 0,%Y,%X
160  PLOT P,%X,%Y;PLOT P,-%Y,-%X;PLOT 0,%Y,%X
170  PLOT P,%X,-%Y
180 PEND
190
```

```
200 FDIM %XX(200),%YY(200)
210 %A=4;T=0
220 P.$12'''''''"          PLEASE WAIT"
230 DO
240  %XXT=(%A*2)*SIN%A
250  %YYT=(%A*2)*COS%A
260  %A=%A+0.1
270  T=T+1
280 UNTIL T=201
290
300 DO
310  C=1;%A=4;T=1
320  CLEAR 4
330  DO
340   BOX(1,T)
350   CLEAR4
360   T=T+1
370  UNTIL T=201
380 UNTIL 0
```

CONDOR

```
 10 PROGRAM CONDOR
 20
 30 REM (C) HAMBURGERSOFT
 40 REM UIT CREATIVE GRAPHICS
 50 REM VOOR DE BBC MICRO
 60
 70 PROC ANDES
 80  FOR J=-50 TO 20 STEP4
 90   MOVE 0,J
100   FOR I=0 TO 25
110    PLOT1,10,(A.R.%20-10)
120   NEXT I
130  NEXT J
140 PEND
150
160 PROC BIRD(X,Y,A,B,V,W,P)
170  X=X/5;Y=Y/5;A=A/5;B=B/5
180  V=V/5;W=W/6-50
190  MOVE(V-A),(W+B)
200  PLOT P,(V-X),(W+Y)
210  PLOT P,V,W
220  PLOT P,(V+X),(W+Y)
230  PLOT P,(V+A),(W+B)
240 PEND
250
260 CLEAR 4
270 ANDES
280 C=1;%A=0
290
300 DO
310  %A=%A+.2
320  C=C%3
330  %V=640+100*COS(%A/12)
340  %W=712+100*SIN(%A/12)
350  %X=(%W-1030)/4
360  %Y=%X*COS(%A)
```

```
370  %C=%X*1.5
380  %B=%C*COS(%A-.4)
390  BIRD(E,F,G,H,I,J,7)
400  BIRD(%X,%Y,%C,%B,%V,%W,5)
410  E=%X;F=%Y;G=%C
420  H=%B;I=%V;J=%W
430 UNTIL 0
```

ELISE

```
10 REM FUR ELISE
20 REM J.G. VAN DEN BRANDHOF
30 REM HUSSELSESTEEG 45
40 REM 3881 LC  PUTTEN
50 GOS.a;PL.A'1;GOS.a;GOS.b;GOS.c;GOS.b;GOS.c;PL.A'2,R4
60 PL.C'4,R4,C'4,R4,C'4,R4,F8,A'8,C'1,F'4.,E'8,E'2,D'2
70 PL.A"#4.,A"8,A"4,G'4,F'4,E'4,D'4,C'4,A'#2,A'2
80 PL.A'#8,A'8,G8,A'8,A'#8,C'1,D'4,D'#4,E'2.,E'4,F'4,A'4
90 PL.C'1,D'4.,B'8,C'8,G'8,G8,A'8,G'8,B'8,G'8,C'8,G'8
100 PL.D'8,G'8,E'8,G'8,C"8,B"8,A"8,G'8,F'8,E'8,D'8,G'8,F'8
110 PL.D'8,C'8,G'8,G8,A'8,G'8,B'8,G'8,C'8,G'8,D'8,G'8,E'8,G'8
120 PL.C"8,B"8,A"8,G'8,F'8,E'8,D'8,G'8,F'8,D'8,E'8,F'8,E'8,F'8
130 PL.E'8,D'#8,E'8,B'8,E'8,D'#8,E'8,B'8,E'8,D'8,E'4.,B'4,E'4
140 PL.D'#4,E'2,B'4,E'4,D'#4;GOS.a;GOS.b;GOS.c;PL.A'1.,R2
150 PL.C'#1.,D'1,E'4,F'4,F'1,F'2,D'#1.,D'1,C'4,A'#4
160 PL.A'1,A'2,A'2,C'2,B'2,A'1.,C'#1.,D'1,E'4,F'4,F'1,F'2,F'1
170 PL.D'#1,D'4,C'4,A'#1,A'2,G#1,G#2,A'2,R4,B'2,R2
180 PL.A8,C8,E8,A'8,C'8,E'8,D'8,C'8,B'8,A'8,C'8,E'8,A"8,C"8
190 PL.E"8,D"8,C"8,B"8,A"#8,A"8,G'#8,G'8,F'#8,F'8,E'#8;GOS.a
200 GOS.b;PL.C4,E4,A'4,B'2,R4,D4,C'4,B'4,C8,A'8,C8,A'8,C8
210 PL.A'1.;E.
220aPL.E'4,D'#4,E'4,D'#4,E'4,B'4,D'4,C'4,A'2,R4,C4,E4,A'4,B'2
230 PL.R4,E4,G#4,B'4,C'2,R4,E4,E'4,D'#4,E'4,D'#4,E'4,B'4,D'4
240 PL.C'4,A'2,R4,C4,E4,A'4,B'2,R4,E4,C'4,B'4;R.
250bPL.A'2,R4,B'4,C'4,D'4
260 PL.E'2.,G4,F'4,E'4,D'2.,F4,E'4,D'4,C'2.,E4,D'4,C'4,B'2
270 PL.R4,E4,E'4,E4,E'4,E4,E'4,D'#4,E'4
280 PL.D'#4,E'4,D'#4,E'4,D'#4,E'4,B'4,D'4,C'4,A'2,R4
290 PL.C4,E4,A'4,B'2,R4,E4,G#4,B'4,C'2,R4
300 PL.E4,E'4,D'#4,E'4,D'#4,E'4,B'4,D'4,C'4,A'2,R4;R.
310cPL.C4,E4,A'4,B'2,R4,E4,C'4,B'4;R.
```

PAGEMODE — 2

```
10 PROGRAM PAGEMODE-2
20
30 REM J. VAN VELZEN
40 REM CTRL-N ALTERNATIEVE
50 REM PAGE-MODE
60 DIM LL10
70 F.N=0TO 10
80  LLN=-1
90 N.
100 P=#2800
110 P.$21;GOS.a;P=#2800;GOS.a;P.$6
120 ?#208=LL0%256;?#209=LL0/256
130 E.
140
```

```
150a[
160:LL00
170       PHP;PHA;TXA;PHA;TYA;PHA
180       LDA#E6
190       CMP@#10
200       BNE LL01
210       JSR#FD69
220       LDA@#01;STA#E6
230:LL01 PLA;TAY;PLA;TAX;PLA;PLP
240       JSR#FE52
250       RTS
260]
270 RET.
```

SQUARE

```
 10 PROGRAM SQUARE
 20
 30 PROC SQUARE(A,B,C),D,E,F,G
 40  XIF C>=1
 50   D=A-C;E=A+C
 60   F=B-C;G=B+C
 70   MOVE D,F
 80   DRAW D,G;DRAW E,G
 90   DRAW E,F;DRAW D,F
100   SQUARE(D,F,C/2)
110   SQUARE(D,G,C/2)
120   SQUARE(E,G,C/2)
130   SQUARE(E,F,C/2)
140  ELSE
150 PEND
160
170 DO
180  INPUT "RADIUS" R
190 UNTIL 0<R AND R<=50
200
210 CLEAR4
220 SQUARE(128,96,R)
230 END
```

DATOMIX

```
 10 PROGRAM DATOMIX
 20
 30 REM PROGRAMMA MADE BY
 40 REM FRENS MEPSCHEN
 50 REM  ZUIDBROEK
 60 REM 05985-2124
 70
 80 DIM DD(13),AA(7),A12
 90 FOR N= 1 TO 13
100   READ DDN
110 NEXT
120 FOR N= 1 TO 7
130   DIM W12;AAN=W
140   READ $AAN
150 NEXT
160
```

```
170 DATA 31,28,31,30,31,30,31
180 DATA 31,30,31,30,31,31
190 DATA "ZONDAG","MAANDAG"
200 DATA "DINSDAG","WOENSDAG"
210 DATA "DONDERDAG","VRIJDAG"
220 DATA "ZATERDAG"
230
240 CLEAR0;P.$30,$15
250 P."               ***********"'
260 P."========== * datomix * ========="
270 P."               ***********"''
280 P."DIT PROGRAMMA BEREKENT EEN"'
290 P."OF MEERDERE VARIABELEN UIT"'
300 P."ONDERSTAANDE REEKS."''
310 P." DAG-DATUM-MAAND-JAAR"''
320 P."VOORWAARDE IS DAT MINSTENS"'
330 P."EEN VARIABELE GEGEVEN IS."'
340 P."VOOR -DAG- :ZONDAG T/M ZATERDAG"'
350 P."VOOR -DATUM- : 1 T/M 31"'
360 P."VOOR -MAAND- : 1 T/M 12"'
370 P."VOOR -JAAR- : 1700 T/M 3000"
380 LINK#FFE3
390 PROC ENTER (V:I)
400   P.$30;VTAB V;HTAB 7
410   ON ERROR I=0;G.430
420   IN.I
430 PEND
440
450 PROC JRTEST
460   IF J<>0;K=J;L=J;G.c
470   IN."VANAF JAAR "K
480   IF K<1700 P.$7;G.470
490   IN."TOT JAAR "L
500   IF L>3000 P.$7;G.490
510   IF L<K;H=K;K=L;L=H
520cPEND
530
540 PROC MNTEST
550   XIF M<>0 N=M;O=M
560   ELSE N=1;O=12
570 PEND
580
590 PROC DTTEST
600   XIF D=0 E=1;F=DD(N)
610     IF N=2 AND K%4=0 F=F+1
620   ELSE E=D;F=D
630 PEND
640
650 CLEAR 0;P.$30;@=2;P=0;W=0;U=0
660 P."               ***********"'
670 P."========== * datomix * ========="
680 P."               ***********"''
690 P." DAG"'"DATUM"'"MAAND"'"JAAR"'
700 P.$30;VTAB 4;HTAB7;IN.$A
710 ENTER(5,D)
720 ENTER(6,M)
730 ENTER(7,J)
```

```
740 JRTEST
750 MNTEST
760 DTTEST;CLEAR0;P.$30
770 IF $A="";GOTO 790
780 GOTO a
790 P=1
800b$A=$AA(P)
810 P=P+1;IF P=8;P=1
820aZ=K/100;Y=K-(Z*100)
830 IF N>2 THEN Q=N;G.f
840 IF Y<>0 Y=Y-1;Q=N+12;G.f
850 Z=Z-1;Y=99;Q=N+12
860fX=%(2.6001*(Q-2)-0.2)+E+Y+Y/4+Z/4-2*Z
870 X=(X%7)+1;IFX <=0 X=X+7
880 IF P<>0 G.o
890 IF $A=$AA(X) G.o
900 GOTO 930
910oU=U+1;W=W+1;IFW=15 W=0;LI.#FFE3
920 P.U" ==> "E"-"N"-"K"-"$AAX'
930 IF E<F;E=E+1;IF E<>F+1;G.p
940 IF N<O;N=N+1;DTTEST;IF N<>O+1;G.p
950 IF K<L;K=K+1;MNTEST;DTTEST;G.p
960 P."NOG EEN BEREKENING (J/N)"
970 IN.$A;IF $A="J";G.650
980 IF $A="N" END
990 P."ALLEEN J OF N"$7$7;G.970
1000
1010
1020pIF P=1 GOTO b
1030 GOTO a
1040 END
```

TUBE

```
10 PROGRAM TUBE
20 REM FRANS VAN HOESEL
30
40 PROC CIRCLE(A,B,Y),X,D,E,F,H,I,J,K,L,M,N
50  X=0;D=3-Y-Y;M=13;N=1;E=B+Y;F=A+Y;H=A;I=B;J=B-Y;K=A-Y
60 DO PLOT M,A,E;PLOT Z,O,Z;PLOT Q,H,E;PLOT R,Z,Z
70 PLOT M,F,B;PLOT Z,O,Z;PLOT Q,K,B;PLOT R,Z,Z
80 PLOT M,A,J;PLOT Z,O,Z;PLOT Q,H,J;PLOT R,Z,Z
90 PLOT M,F,I;PLOT Z,O,Z;PLOT Q,K,I;PLOT R,Z,Z
100    IF D<0 D=D+4*X+6;G.c
110      D=D+4*(X-Y)+10;Y=Y-N;E=E-N;F=F-N;J=J+N;K=K+N
120c     X=X+N;A=A+N;B=B+N;H=H-N;I=I-N
130  U.X>Y
140 PEND
150
160 CLEAR4
170 MOVE0,0;DRAW0,191;DRAW255,191;DRAW255,0;DRAW0,0
180 Z=0;O=-1;R=9;Q=7
190 X=30;Y=30
```

```
200 DO I=A.R.%30
210aDOV=R.%6;W=R.%6;I=A.R.%30
220 U.X+V>15 AND X+V<234 AND Y+W>15 AND Y+W<166
230 DOX=X+V;Y=Y+W
240  CIRCLE (X,Y,12)
250  KEYK;IF K<>0 INKEYK
260  I=I-1
270  U.I<0ORX<22ORX>233ORY<22ORY>165
280 U.0
```

CIRCLE

```
 10 PROGRAM CIRCLE
 20 REM ALGORITHME BY
 30 REM  J.MICHENER
 40
 50 PROC CIRCLE(A,B,R),X,Y,D
 60  X=0;Y=R
 70  D=3-2*R
 80  DO PLOT 13,(A+X),(B+Y)
 90      PLOT 13,(A+Y),(B+X)
100      PLOT 13,(A+Y),(B-X)
110      PLOT 13,(A+X),(B-Y)
120      PLOT 13,(A-X),(B-Y)
130      PLOT 13,(A-Y),(B-X)
140      PLOT 13,(A-Y),(B+X)
150      PLOT 13,(A-X),(B+Y)
160      XIF D<0 D=D+4*X+6
170      ELSE D=D+4*(X-Y)+10;Y=Y-1
180      X=X+1
190  UNTIL X>Y
200 PEND
210
220 CLEAR4
230 FOR R=7 TO 64 STEP 4
240  CIRCLE(127,96,R)
250 NEXT
260 END
```